sinclair COMPUTER

Mensile per gli utenti di ZX 81 e Spectrum

#1

L. 2.500



Anno 1 - N° 1 - Febbraio 1984 - Sped. Abb. Post. Gr. III/70

a casa tua subito?

SI

- Tutti i videogames e i programmi per computer puoi riceverli a casa tua.

- Non è incredibile?

E invece è proprio cosi: dovunque tu abiti, dovunque ti trovi puoi avere nelle tue mani in pochissimo tempo tutti i videogiochi che vuoi.

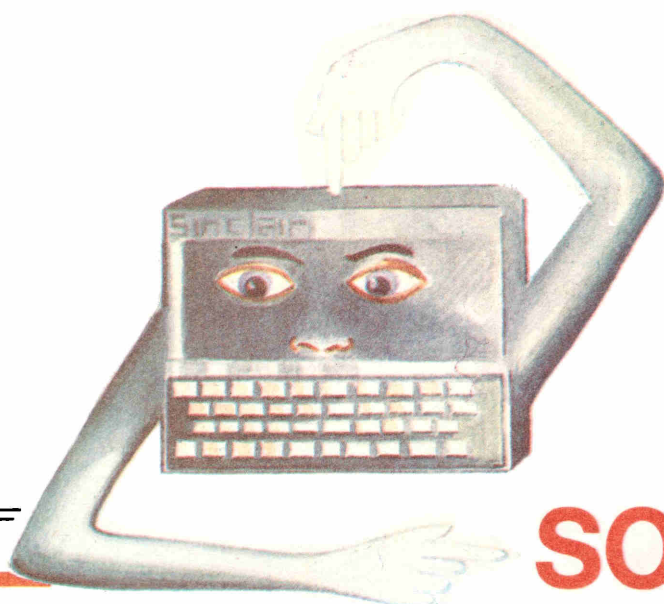
- Questo è lo straordinario servizio offerto da EXELCO, la grande Compagnia specializzata in spedizione e recapito di materiale informatico ed elettronico.

- Dopo pochi giorni il postino suonerà - tre volte - alla tua porta, verificherai l'esattezza della merce spedita — EXELCO non ha mai sbagliato, ma tu verifica in ogni caso — e tu pagherai i tuoi videogames.

- Comodo, vero?

COMODISSIMO EXELCO

l'unica organizzazione europea per la spedizione di videogames



sinclair COMPUTER

SOMMARIO

DIRETTORE RESPONSABILE
Agostina Ronchetti

COORDINAMENTO TECNICO
Mauro Soldavini

REDAZIONE
Lorenzo Bossi
Franco Raspa

GRAFICA, IMPAGINAZIONE E DISEGNI
Renato Caruso
Francesco Amatori

FOTO
Franco Vignati

DIFFUSIONE E ABBONAMENTI
Marina

DIREZIONE, REDAZIONE

Via Cristoforo Colombo, 49
20090 Trezzano/Milano - Tel. 02/8466675

Autorizzazione del Tribunale di Milano
n. 522 del 12.11.83

STAMPA:
Pizzorni (Cremona)

Concessionario esclusivo per la diffusione:
MEPE spa - Via G. Carcano, 32 - Milano

Spedizione in abbonamento postale
Gruppo III/70

Prezzo della rivista L. 2.500
Numero arretrato L. 5.000

Abbonamento annuo L. 20.000.
I versamenti vanno indirizzati a: Sinclair Computer - V.le Famagosta, 75 - 20142 Milano - mediante emissione di assegno bancario, cartolina-vaglia o utilizzando il c/c postale
n.ro 3042609

Per i cambi di indirizzo, indicare, oltre naturalmente al nuovo, anche l'indirizzo precedente, ed allegare alla comunicazione l'importo di L. 500 anche in francobolli.

sinclair COMPUTER

ti offre un... invito a collaborare



Quando inviate il vostro software...

... per favore cercate di attenervi il più possibile alle seguenti...

... indicazioni per la presentazione di un programma.

1. *Descrizione* di che cosa fa e come si usa.
2. *Commento* su particolari procedure usate, se pensate di aver fatto qualcosa di originale; eventuali possibili modifiche.
3. *Elenco delle variabili usate* (almeno nel caso di programmi complessi, calcoli tecnici, formule di uso non comune).
4. *Listato del programma da stampante,* in versione ben collaudata e funzionante; se non avete la stampante, mandate il list dattilografato, ma in questo caso *la cassetta con il programma registrato è indispensabile* (è comunque sempre ben accetta: caricare i programmi dalla tastiera rende i nostri collaboratori particolarmente aggressivi).
5. *Includete qualche "videata"* (hard-copy dello schermo) significativa, se la ritenete opportuna per mostrare che cosa fa il programma.

Il tutto cercando di *non superare* le tre cartelle dattiloscritte di testo (list escluso); romanzi più lunghi saranno pubblicati solo in caso di software molto interessanti e complessi. Quindi, tagliate senza pietà tutte le divagazioni non pertinenti.

sinclair*amentevostro*



Chi scrive e chi legge

Questa è (meglio: sarà) la rubrica della posta, che speriamo sarà sempre numerosa, scritta a macchina (ma va bene anche a penna, con la biro o la matita o la stampante), con nome, cognome e indirizzo del mittente. Potete inviare anche disegni, programmi, schizzi e frittelle. I francobolli per le risposte particolari non ci servono: infatti, risponderemo solo nelle rubriche. Anzi, non risponderemo quasi mai, perchè ci piacerebbe che le risposte nascessero dall'accostamento di più lettere dei lettori. Ma se ciò non sarà possibile, vuol dire che risponderemo noi.

Per facilitare la lettura della corrispondenza, la rubrica verrà divisa in più sezioni, organizzate come segue.

HELP

Questa è la sezione di chi chiede aiuto: come si fa per...? Ho fatto così e cosà, ma il mio Sinc s'inchioda, come debbo schiodarlo? E così via, capito?

Cercheremo sempre di darvi la risposta giusta.

EUREKA

Questa sezione non avrà bisogno di risposte. Infatti raccoglierà le lettere di quanti hanno fatto scoperte piccole e grandi, relative all'uso delle nostre macchinette.

Naturalmente, daremo la precedenza a chi scoprirà le leggi della gravità dei computer (gettandoli dalla stessa altezza cadono tutti con la stessa velocità, ma non tutti si fracassano dopo lo stesso numero di cadute) o a chi inventerà nuovi linguaggi e sistemi operativi. Pensiamo di non dovere riservare troppo spazio per chi si limiterà ad inventare solo l'acqua tiepida.

Approvo/disapprovo

In questa sezione verranno raccolte, invece, lettere di contestazione contro i redattori della rivista, i giornalai, i rivenditori di computer, i loro amici e compari e - speriamo - anche lettere di encomio e solidarietà, oppure prese di posizione - mettiamo - con chi sostiene che tutti i Sinclair al buio sono neri e chi ritiene, invece, il contrario.

Sapete che?

Se qualcuno vi dice (o leggete su una rivista inglese) che è possibile fare certe cose, ma non l'avete scoperto voi (quindi rientra Eureka) questo è l'angolo dove raccontarlo a tutti gli altri.

RANDOM

Naturalmente, se scriverete altre cose, troveremo altri modi per incasellare le vostre lettere. E, se non è facile, le metteremo così alla rinfusa, ossia in modo RANDOM, per dirla con lo zio Clive.

Sinclairamente Vostro

Clive Nipote

per questa rubrica scrivere a
Sinclair Computer / posta
casella postale ....
.... Brescia

sinclairama



QL: un nuovo standard

Ancora una volta Sir Clive Sinclair sconvolge tutti gli ordini di grandezza col suo atteso personal professionale, destinato a rimettere in discussione tutti gli standard di questo settore.

Si chiama QL, dalle iniziali inglesi di Quantum Leap, ossia "salto di parecchi ordini di grandezza".

Rivolto alle applicazioni professionali più che hobbistiche, il QL di Sinclair utilizza un microprocessore Motorola 68008 con struttura interna a 32 bit. La sua capacità di memoria è di 128 Kb e possiede due microdischi integrati da 180 Kb ed una vera tastiera. Nove porte convertono il collegamento con le periferiche e la comunicazione secondo gli standard abituali, ivi compresa una rete locale.

Il sistema operativo Q-DOS permette l'utilizzazione multielaborazione (ma monoposto) e può visualizzare fino a 20 elaborazioni simultanee ("finestre" multiple).

Il linguaggio, battezzato Super-Basic è una versione migliorata del basic dello Spectrum.

Il sistema viene offerto con un pacchetto di elaborazione testi (Q-Quill), un foglio elettronico (QL-A bacus), un sistema di gestione di base dati (QL-Archive) ed un package grafico (Q-Easil) sviluppato dalla Psion.

In Inghilterra il QL verrà venduto a partire dal mese prossimo a 399 sterline, ossia l'85% meno del personal Ibm e l'82% meno dell'Apple IIe. La pretesa di QL di "cambiare gli ordini di grandezza del settore" è, dunque, giustificata.

Secondo il direttore generale della Sinclair, le vendite di QL dovrebbero raggiungere rapidamente i 100 mila pezzi al mese.



Il QL (leggi Qu-éll) è tutto qui. Nella foto si vede anche il doppio microdrive e la cartuccia da O.5MB.
In Italia l'esclusivista Rebit lo venderà intorno a lire 1.300.000 + Iva.



TOP 10

Arcade - games Spectrum

1. Flight simulation (*Psion / Sinclair*)
2. Chequered flag (*Psion / Sinclair*)
3. Manic miner (*Bug / Byte*)
4. 3D - Panxs (*DK' Tronics*)
5. Escape (*Rebit*)
6. Naziacs (*DK' Tronics*)
7. Horace and the spiders (*Psion / Sinclair*)
8. Monst in hell (*Softek*)
9. Acquarius (*Bug / Byte*)
10. Horace goes sking (*Psion / Sinclair*)

Giochi da tavolo e di strategia

1. Chess (*Psion / Rebit*)
2. Reversi (*Psion*)
3. IGK superchess (*CP Software*)
4. Planet of death (*Sinclair*)
5. ZX daught (*CP Software*)

Utility

1. VU - Sile (*Psion / Rebit*)
2. VU - Calc (*Psion / Rebit*)
3. Editor Assembler (*Picturesque*)
4. Snail Logo (*CP Software*)
5. Word Processing (*Rebit*)

Per questa rubrica scrivi a:
Sinclair Computer
c/o Sigma Informatica
V. Cristoforo Colombo, 49
20090 Trezzano
Milano

La prima tappa d'un appassionato viaggio nell'architettura della macchina.

Viaggio nello Spectrum



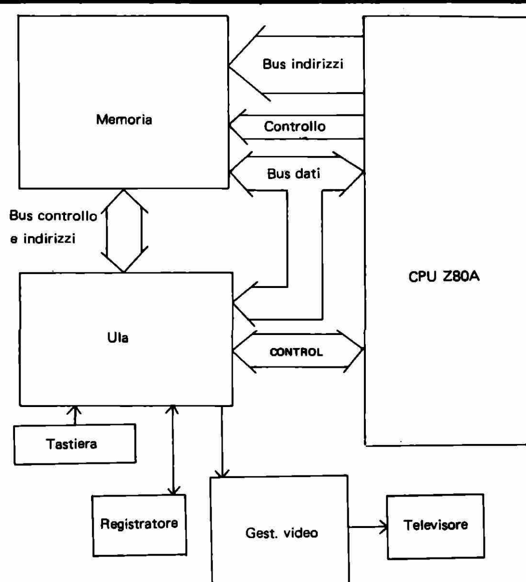


Fig. 1. Il diagramma a blocchi dello Spectrum

NELLE descrizioni seguenti delle diverse sezioni che compongono lo Spectrum, fate riferimento al diagramma a blocchi generale della fig. 1.

L'unità di elaborazione principale, comunemente chiamata CPU (= central processing unit) è, come dice il nome, la centrale operativa del computer. E' collegata alle altri parti dai BUS, di cui parleremo diffusamente più avanti; ci sono bus per dati, bus di controllo e bus di indirizzi.

La CPU dello Spectrum, è lo Z80A, il grosso chip siglato IC2, al centro della scheda (fig. 2). Questo micro-processore è un dispositivo a 8 bit: ciò significa che ci sono otto differenti connessioni nel suo bus dati. Attraverso questo, la CPU può inviare informazioni agli altri dispositivi, e viceversa.

Poichè ci sono otto connessioni, ciascuna delle quali può trovarsi in stato logico 0 o 1, attraverso il

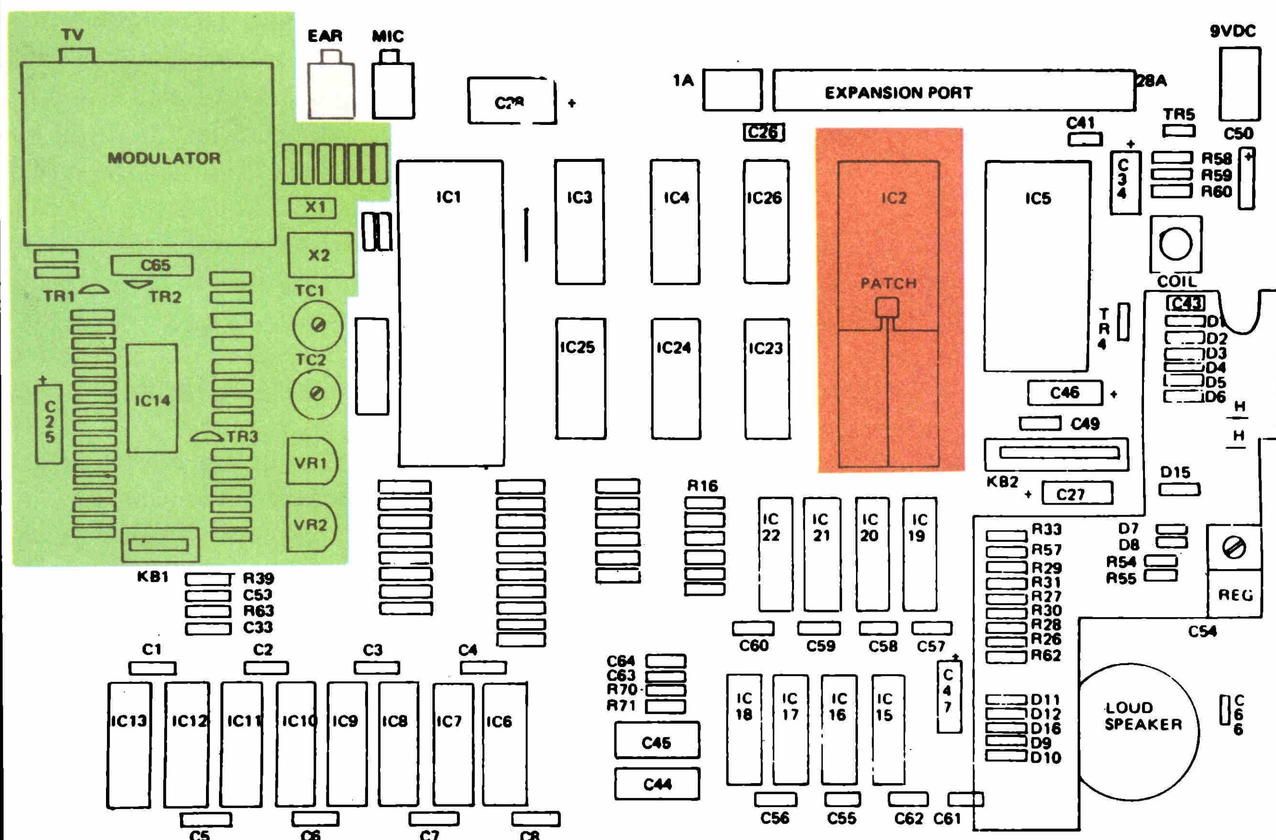


Fig. 2. La scheda dello Spectrum, serie 2.

bus dati, possono essere inviati numeri compresi tra 0 (tutte le porte in stato 0) e 255 (tutti 1); la rappresentazione binaria di 255 è, infatti, 1111 1111. A questo punto, potreste chiedervi come fa la CPU a riconoscere un numero molto grande o una parola digitata sulla tastiera in basic. Potrebbe sembrare difficile capire come si possa rappresentare il vostro nome, battuto sui tasti dello Spectrum, solo con numeri compresi tra 0 e 255. La risposta è più semplice di quanto pensiate.

La CPU procede eseguendo una piccola parte dell'operazione per volta. Per capire una parola come BOBO, inizia con la "B" (che vede come 66 decimale), quindi analizza la "O" (79 decimale), etc. L'elenco completo dei caratteri con gli equivalenti decimali, lo troverete nell'appendice A del manuale dello Spectrum [1], oppure potete listarli sul video così:

```
10 FOR K = 32 TO 255
20 PRINT K, CHR$ K
30 NEXT K
```

I grandi numeri decimali vengono trattati in modo simile. Ogni decimale è memorizzato nello Spectrum in 5 bytes di memoria (vedi per maggiori chiarimenti il cap. 33 del manuale).

Per iniziare a fare qualcosa, la CPU deve essere istruita su che cosa fare. Le istruzioni per far girare il basic sono conservate stabilmente nella memoria. Il programma per il sistema operativo del basic contiene tutte le informazioni richieste dalla CPU per "capire" il basic stesso. Questo programma è, ovviamente, scritto in codice macchina, e inizia a girare non appena si accende il computer. Si trova memorizzato nel chip di ROM siglato IC5. La ROM (= read only memory) è "memoria di sola lettura", non modificabile dalla CPU nè dall'utente, e il programma è conservato nel chip anche quando si spegne la macchina.

Le istruzioni che introducete nello Spectrum, vanno a disporsi nella memoria ad accesso casuale (RAM = random access memory).

A differenza della ROM, questa memoria può essere modificata dalla CPU e perde tutte le informazioni che contiene quando si spegne la macchina; per conservare i dati contenuti in RAM, occorre trasferirli su una cassetta - o comunque, su un supporto magnetico.

Trovato che cosa fare del programma residente in ROM, la CPU deve ricevere input dalla tastiera o da cassetta, e invierà output al video o alla cassetta. La ULA (= uncommitted logic array) aiuta la CPU a interfacciarsi con il mondo esterno. La ULA prende le informazioni direttamente dagli input di tastiera o di cassetta, e le invia alla CPU. Quando questa vuole registrare un programma, o far suonare il beeper, "dice" alla ULA di farlo. L'output al televisore è un po' più complicato: la ULA copia l'output di schermo, dalla memoria video al circuito di uscita video, 50 volte al secondo: ciò dà l'impressione di un'immagine continua. Quando la CPU deve inviare al televisore un'immagine, non fa che trasferire le informazioni da mostrare nella memoria video. La ULA fa il resto.

Vediamo ora come avviene il trasferimento delle informazioni tra le diverse parti del computer. Tutti i dati vengono movimentati attraverso il bus dati. Il tipo di trasferimento che deve essere eseguito è definito da vari segnali di controllo, inviati sul bus di controllo. Per esempio, la CPU invia un codice di lettura, se deve leggere dei dati dalla ULA o dalla memoria. Ciò avvisa la ULA o la memoria di inviare quei dati alla CPU. Allo stesso modo ci sono codici di scrittura.

Per indicare alla memoria dove scrivere i dati, la CPU fornisce, al bus indirizzi a 16 bit, un indirizzamento.

Il bus indirizzi consente di leggere/scrivere dati in 2^{16} = 65.536 diverse locazioni di memoria.

I bus possono soltanto lavorare con gli stati logici "0" e "1" - su ciascuna linea. Nella pratica, questi stati logici sono rappresentati da diversi livelli di voltaggio. Per convenzione - per rendere compatibili tra loro tutti i moderni circuiti integrati per computer - lo stato logico "0", corrisponde a una tensione tra 0 e 0.8V, lo stato logico "1" tra +2 e +5V (massima tensione di alimentazione).

Quando la tensione è intermedia - tra 0.8 e 2 -, il segnale sta cambiando da 0 a 1 (o viceversa). Tutti i trasferimenti di dati avvengono entro questi

limiti di tensione. I chip sono progettati in modo da non tenere conto dei dati nei momenti in cui la tensione sta cambiando.

L'alimentazione

L'alimentatore di un computer è, probabilmente, la parte di circuitazione più spesso trascurata dagli utenti. Si fa il collegamento alla rete, e il computer comincia a vivere. Si dà per scontato che vengano generate la giusta corrente e la giusta tensione. Tuttavia, se volete mettere le mani, o almeno la testa, nell'hardware del vostro Spectrum, è necessaria una buona conoscenza delle caratteristiche dell'alimentazione.

Tutta l'energia fornita entra alla tensione di +9V, con una corrente massima di 1.2A (confrontate la targhetta sull'alimentatore). Ora, si dà il caso che nessuno dei chips usi +9V. La maggior parte dei circuiti logici, CPU compresa, richiede +5V. La ULA ne vuole +12. I chip di memoria video sono più complessi, e pretendono +12V, +5V e −5V contemporaneamente. Quindi il problema non è soltanto quello di produrre un'alimentazione più o meno costante per la maggior parte del tempo, e più o meno allo stesso voltaggio. I +5V hanno una tolleranza del 5% e i +12V e −5V possono sgarrare del 10% dal valore nominale su tutto il circuito in qualsiasi momento. Ma una caduta di tensione di un solo microsecondo, può provocare un disastro (per il software, naturalmente). Come si possono produrre tensioni costanti?

+5 Volt

Questa è l'alimentazione principale. Nello Spectrum 48k è sfruttata al massimo, dovendo fornire 1 intero Ampère. Il piccolo regolatore +5V (fig. 2) è avvitato su una larga piastra di alluminio: è un integrato con tre piedini, al margine destro della scheda.

Questo regolatore in tensione, siglato 7805 (fig. 3), riceve +9V al pin IN e restituisce +5V stabilizzati al pin OUT. La tolleranza della tensione in entrata è alta (da +7V a +25V) e verranno sempre restituiti +5V, rigorosamente costanti; soltanto ci sarà più o meno dispersione. Questa avviene sotto forma di calore, attraverso il dissipatore di alluminio. Ecco perchè la parte destra dello Spectrum diviene molto calda dopo un po' che è acceso(2).

+12 Volt

Per avere +5V, partendo da 9, è abbastanza facile: se ne tolgono 4. Per averne +12, sempre partendo da 9, è un po' meno semplice.

I transistor TR5 e TR4 (fig. 4) e i componenti a loro collegati, assolvono a questa funzione. TR5 genera un feedback di corrente per l'oscillatore formato da C43, R61, C1 e TR4 (che è il transistor che controlla l'alimentazione principale). Le operazioni del circuito si basano sulla tensione inversa indotta che attraversa L1, il che accade ad ogni ciclo di oscillazione. La tensione inversa manda il collettore di TR4 dai +9V a un massimo di circa +13V. A questo livello, D15 conduce e carica il condensatore C44, il quale, scaricandosi quando D15 non conduce, fornisce una tensione costante di +12V ai circuiti di ULA, memorie e video. Se la tensione tende ad abbassarsi, TR5 inizia a condurre di più, il che fa aumentare la frequenza dell'oscillatore, rialzando il voltaggio al livello originario.

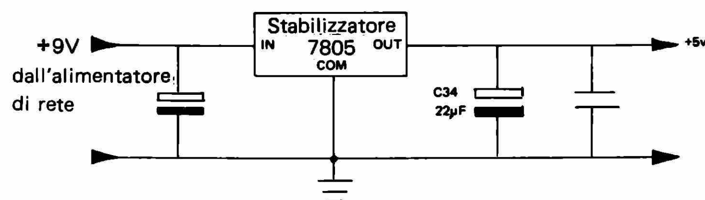


Fig. 3. Il circuito di alimentazione + 5V

–5 Volt

La sezione circuito che si occupa dei –5V, consiste di C46, D11, R55, D12, D17, R54, C47 (fig. 4). Funziona col principio della "pompa di alimentazione". Abbiamo visto che il collettore di TR4 nel circuito +12V oscilla rapidamente tra circa +13V e 0V. Quando arriva a +13V, C46 si carica (attraverso D11) fino a un massimo di circa +12V (0.7V vengono dissipati nel passaggio attraverso i diodi al silicio). Quando il collettore di TR4 torna a zero, il polo negativo di C46 scende a –12V. Quindi, C47 si carica, attraverso D12 e R55, da C46. La tensione in C47 è tenuta costante a –5V dal diodo Zener D17. C46 viene poi ricaricato dal successivo ciclo dell'oscillatore, e così via.

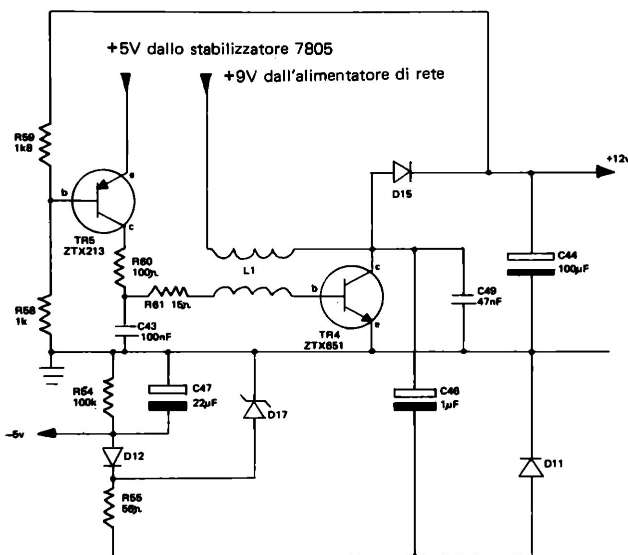


Fig. 4. Il circuito di alimentazione + 12V e -5V

Limiti di corrente

Lo Spectrum può fornire corrente dai propri

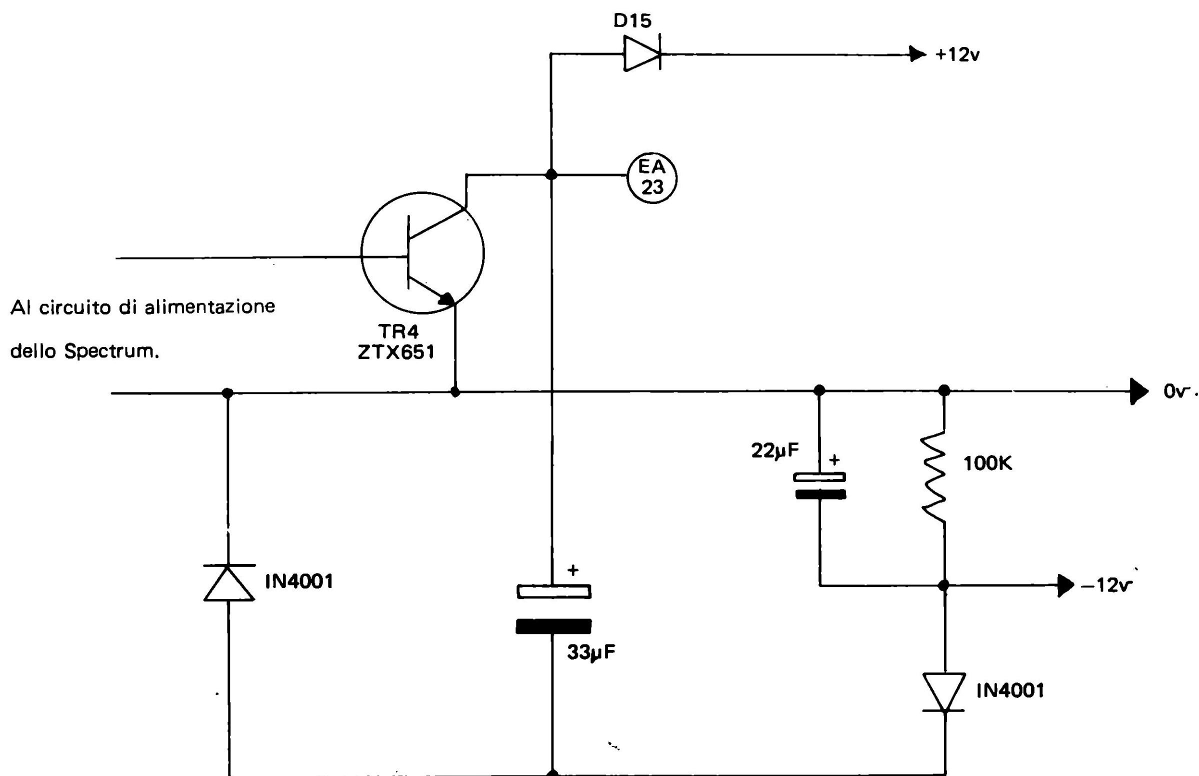


Fig. 5. Circuito supplementare per alimentazione -12V

circuiti di alimentazione a circuiti esterni, ma tale corrente è molto limitata. Dal "16k", si possono trarre circa 300mA dalla linea +5V. Il limite massimo assoluto è difficile da definire. In genere, più togliete corrente e peggio funziona la stabilizzazione.

Una cattiva stabilizzazione può causare frequenti "crash" del sistema (con "crash" intendiamo tutti quegli inconvenienti che fanno perdere il controllo del computer; gli effetti esteriori sono: la tastiera che va "in palla", il video che si oscura o dà segnali casuali, la stampante che impazzisce: l'unica soluzione, in genere, è togliere l'alimentazione e cominciare da capo, poichè, ovviamente, si perde il programma), ma nessun danno permanente se si rimane nei limiti indicati prima. Con il 48k è opportuno aggiungere un'alimentazione supplementare a +5V per tutti i circuiti esterni.

–12 Volt

Ed eccoci ai –12V: questa tensione è indicata a un piedino inferiore del connettore di uscita (v. cap. 35 del manuale), però non la si trova nello schema generale del circuito. Infatti, non esiste.

Il piedino –12V è stato etichettato in maniera inesatta: si tratta, in realtà, di "+12V non stabilizzati", visto che è collegato al collettore di TR4.

Fortunatamente, è abbastanza facile usare questi +12V oscillanti per produrre –12V reali, con l'aggiunta di pochi componenti comuni. Nella fig. 5, potete riconoscere un circuito molto simile a quello presente nello Spectrum per i –5V. Lavora esattamente con lo stesso principio della pompa di alimentazione. Non c'è il diodo Zener, perchè il condensatore da 33 microF si carica in ogni caso a 12V, ed è, quindi, già alla tensione giusta. La stabilizzazione di questo circuito, non è mai molto buona sotto carico: può arrivare a –10.5V se prelevate 20mA dal circuito, e questo è il massimo di corrente raccomandato.

Se costruite il circuito, assicuratevi che i due condensatori siano tarati per almeno 16V. A parte ciò, nessuno dei componenti ha valori critici.

+5 Volt extra

Per compiere semplici esperimenti con lo Spectrum 48k, usando circuiti addizionali, è utile avere a disposizione un'alimentazione supplementare esterna; per il 16k abbiamo visto che si può disporre di 300mA, ma non sempre sono sufficienti. In

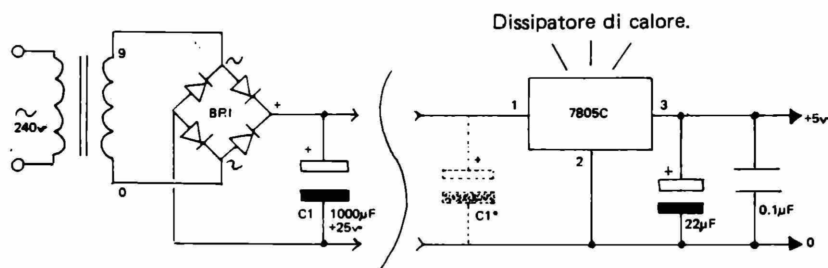


Fig. 6. Alimentazione addizionale + 5V

ogni caso, avete due possibilità:

1/ Usare l'alimentatore della Sinclair. Il trasformatore a +9V ha qualche disponibilità di corrente extra, specialmente se la linea +12 V non è a pieno carico. Ci sono circa 500mA e, per sfruttarli, dovete approntare il semplice dispositivo di fig. 6, limitatamente alla parte destra del disegno. Lo schema ricalca il circuito che si trova dentro lo Spectrum.

Collegate lo stabilizzatore 7805 come mostrato in fig. 7, con la linea +9V al pin di INPUT. Il regolatore è mostrato dal lato della plastica nera, su cui è stampata la sigla. Montate il 7805 su un dissipatore di calore: si trovano facilmente dei supporti alettati nei negozi di componentistica. Oppure usate una piastra di alluminio grande come quella del densatore di livellamento (C1=1000microF/+25V). Assicuratevi che il trasformatore abbia un buon isolamento; potete montarlo in una scatola metallica, collegata a terra, che potrà funzionare anche da dissipatore per il 7805. Curate anche il buon isolamento del collegamento alla rete.

Dovete fare attenzione che tutto il sistema vada in tensione nello stesso istante: se ciò non avviene, qualche chip può restarne danneggiato; non gradiscono, infatti, di ricevere tensione a qualche pin, senza che tutto il circuito sia alimentato.

Condensatori di disaccoppiamento

Con la messa in tensione, l'energia è distribuita su tutta la scheda. Tuttavia, se un chip, sul lato opposto della scheda rispetto all'alimentazione, improvvisamente richiede più corrente, può verificarsi un abbassamento locale di tensione. Se si abbassa troppo per un solo microsecondo, può essere sufficiente per causare la perdita di dati e programmi in memoria. Per questo motivo, si piazzano i condensatori di disaccoppiamento, spargendoli in posizioni strategiche sulla scheda. Essi sono in grado di stabilizzare ulteriormente la tensione, fornendo correnti forti per brevi periodi di tempo. Condensatori con questa funzione sono quelli da C1 a C8, presso i chip di memoria. ■

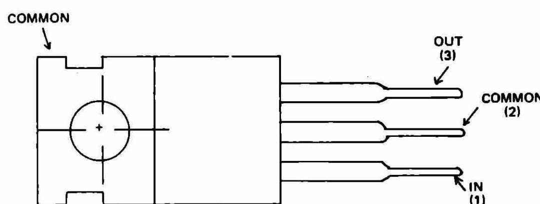


Fig. 7. I piedini dello stabilizzatore 7805 1A/5V (visto dall'alto)

computer. Il condensatore C1 di fig. 6 è facoltativo, ma aiuta la stabilizzazione: 22microF/16V. Da 22microF è anche l'altro condensatore; è opportuno anche distribuire qualche piccolo condensatore di disaccoppiamento da 0.1microF: uno ogni due chip.

I +5V forniti da questo circuito non devono essere collegati alla linea +5V dello Spectrum: soltanto la connessione OV andrà fatta a tutti i chip.

2/ Usare un alimentatore supplementare. Il circuito è ancora quello di fig. 6, questa volta compresa la parte sinistra. Fornisce circa 1A a +5V. Nell'ordine, troviamo un trasformatore 220/9 da 1A, un raddrizzatore a ponte da 1A (BR1) e un grosso con-

(1) Qui, e per tutto il testo, si fa riferimento al manuale italiano dello Spectrum, a cura di R. Bonelli, edito dalla JACKSON.

(2) In verità, diventa anche troppo calda; ciò è dovuto ad una tensione un po' alta, fornita da molti alimentatori (parecchio maggiore di +9V) e alla scarsa aereazione. Normalmente non ci sono rischi, ma vedremo come migliorare il raffreddamento.

"Microdrive"

.... Tutto quello che c'é da sapere sullo ZX -Microdrive

Presentato ufficialmente in ottobre, non sarà commercializzato in Jtalia che in primavera

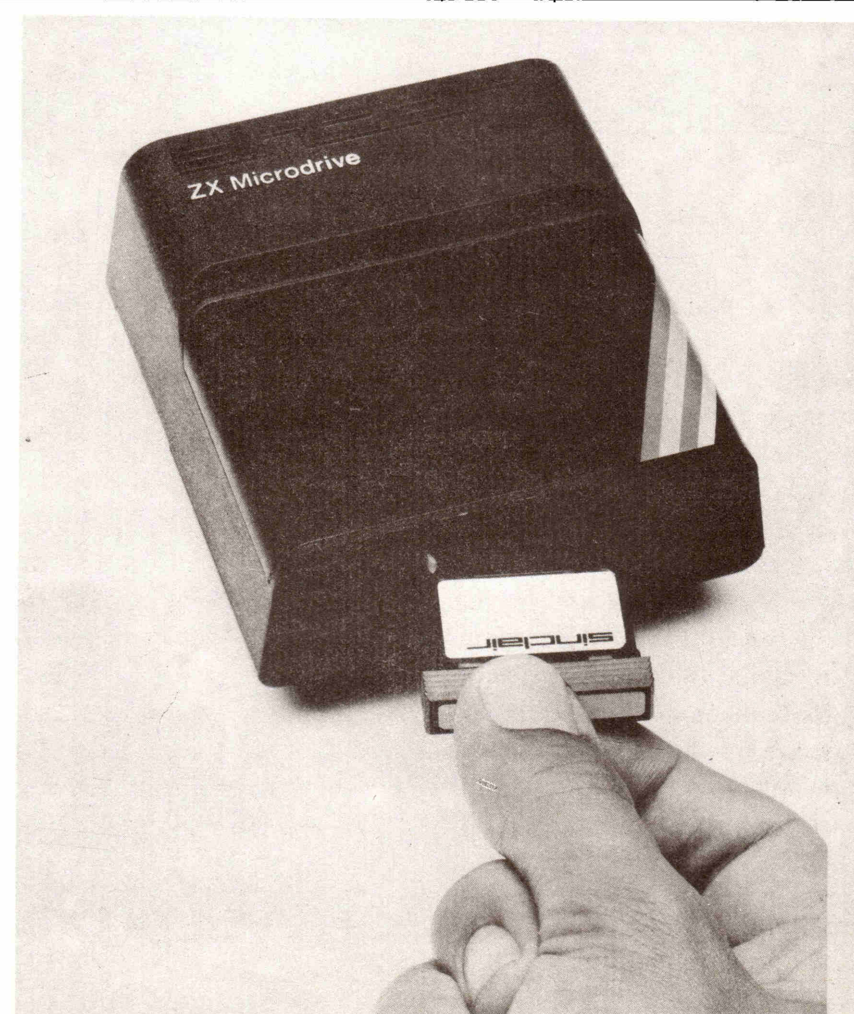
PER qualche sinclairista che proprio non sapesse ancora di che cosa stiamo parlando, il "Microdrive" è la nuova unità di memoria di massa destinata allo Spectrum. Annunciata inizialmente come un piccolo drive per mini-mini-floppy-disk, da 3" o giù di lì, quando è stata finalmente presentata, sul finire dell'estate, si è rivelata una nuova sorpresa dell'inesauribile Oncle Clive (zio Clive, e che sinclairisti siete, se non sapete chi è?): al posto del dischetto c'è una cartuccia di nastro magnetico, avvolto ad anello senza fine.

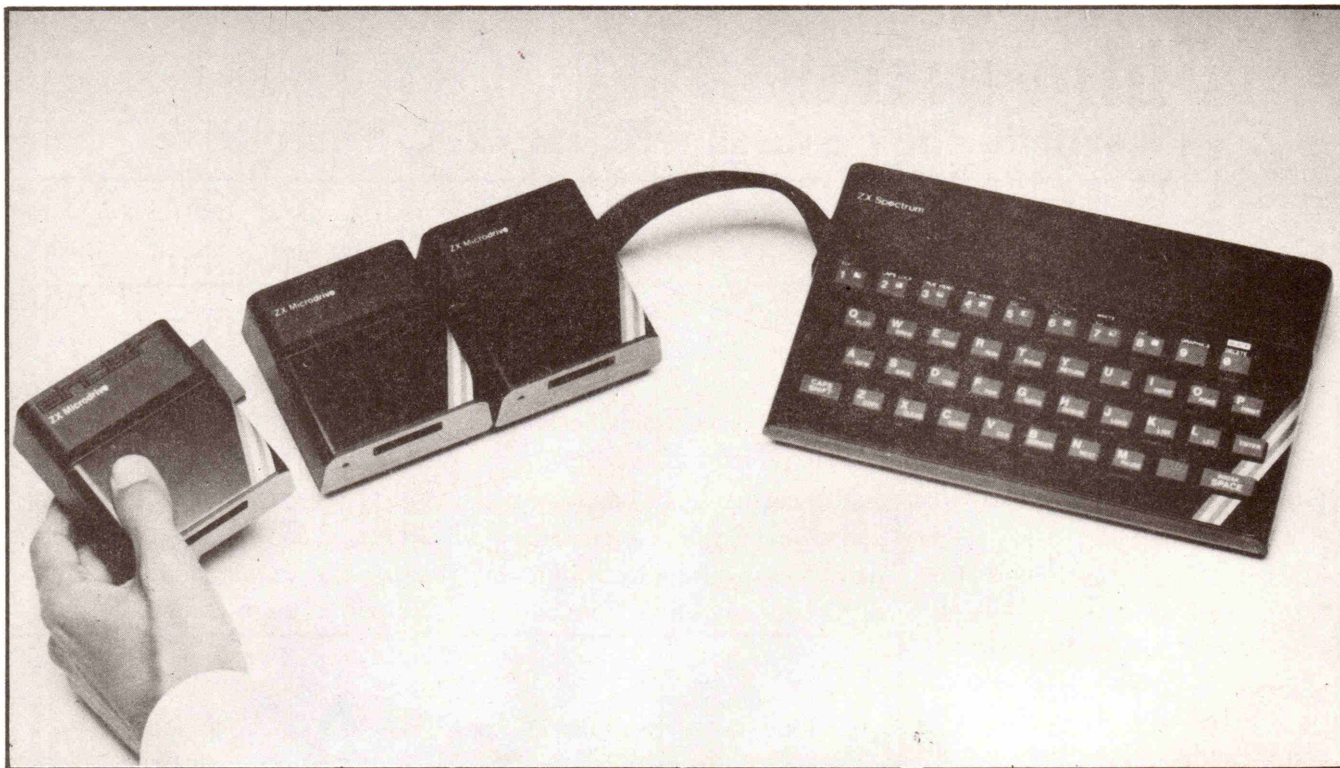
Lo ZX-Microdrive si presenta come una scatoletta di circa 9x9x5 cm, con un design che ricalca quello dello Spectrum, e pesa un po' meno di 250 g. Per collegarlo al computer occorre una scheda di interfaccia che si infila, ovviamente, nell'edge-connector posteriore e si va a posizionare sotto lo Spectrum stesso. Posizione quanto mai felice, perchè lo solleva un po' dal tavolo, migliorando sia l'ergonomia della tastiera, sia il raffreddamento.

Questa scheda, denominata INTERFACE 1 (dell'Interface 2 parliamo dopo), ha diverse uscite; riporta il connettore per altre espansioni, ha due prese EAR/MIC come il computer, la presa apposita per il cavo piatto di collegamento del Microdrive, e infine, una RS232. Di unità a nastro se ne possono collegare più di una, ponendole in serie.

La cartuccia di nastro, da inserire nell'apposita fessura del drive, è una cosina incredibilmente piccola - 3x4.5x0.7 cm., accreditata di una capacità di 85k.

Utilizza, pare, nastro magnetico da videoregistrazione, alto pochi mm. e la testina di lettura/scrittura assomiglia molto ad una normale, da registratore a cassette. L'inserimento della cartuccia aziona un microswitch, avver-



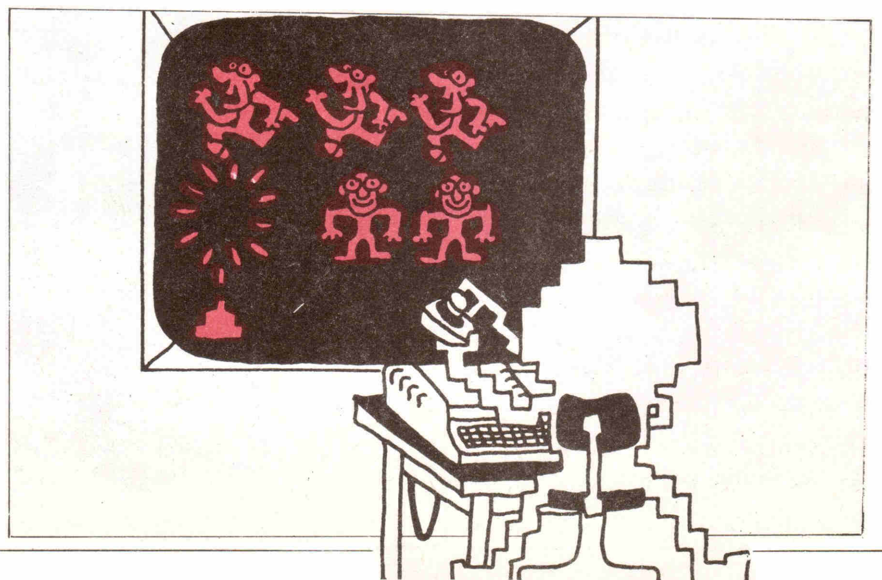


tendo il sistema che il drive è operativo; come nei disk-drive, un led rosso si accende, ad indicare "lavori in corso", quando il nastro sta girando.

Sempre al pari dei disk, un nastro nuovo richiede una inizializzazione, che dura una trentina di secondi. Il tempo di trasferimento dei dati è molto buono: un programma di media lunghezza viene memorizzato in 3-4 secondi netti, più un periodo variabile da zero a un'altra decina di secondi, dipendente dal grado di riempimento della cartuccia: se è già quasi satura, serve un po' per trovare dove vi sia ancora spazio libero.

Il LOAD del nastro è leggermente più lento del SAVE, e il trasferimento di files DATA è più lento di quello dei programmi, ma resta comunque molto veloce: l'apertura di un file, la scrittura di 50k di dati e la chiusura, richiedono poco più di un minuto (diamo credito a quanto pubblicato dalle riviste inglesi, che hanno potuto effettuare i primi test).

Un'altra caratteristica molto interessante dell'interfaccia: permetterebbe a due o più Spectrum (abbiamo letto fino a 64, ma non sappiamo se crederci) di collegarsi tra loro. Non viene però chiarito che genere di dialogo venga abilitato (forse che nessuno ha ancora provato? !): aspettiamo di saperne di più.



W l'algebra
M l'algebra

Massimo Rossi

Due brevi routines per lo zx81 e lo Spectrum. Per rendere più agevoli gli esercizi di algebra.

IL primo programma, per lo ZX81, calcola le radici di un'equazione di secondo grado, un esercizio che al liceo capita di risolvere qualche migliaio di volte. Se la parte della formula risolutiva, che si trova sotto radice è negativa (dipende dal valore dei coefficienti), non si hanno soluzioni reali: da un numero negativo non si può estrarre la radice quadrata; la matematica prevede, a questo, punto delle soluzioni immaginarie, che nel programma vengono segnalate e lasciate agli addetti ai lavori.

Il secondo programma di matematica calcola il massimo di una funzione in un intervallo definito. Occorre avere un'idea almeno approssimativa dell'andamento della funzione in quell'intervallo, per evitare che vi si trovino due picchi: il programma ne individuerebbe soltanto uno, senza riuscire a stabilire se è effettivamente il maggiore.

La routine è anche una semplice dimostrazione dell'uso dell'istruzione DEF FN in dotazione allo Spectrum, il cui uso è a volte oscuro ai neofiti. Dato il RUN, viene richiesta la funzione che si vuole studiare, una qualsiasi y=f(x) lineare, da introdurre come linea di programma (il quale viene ovviamente fermato).

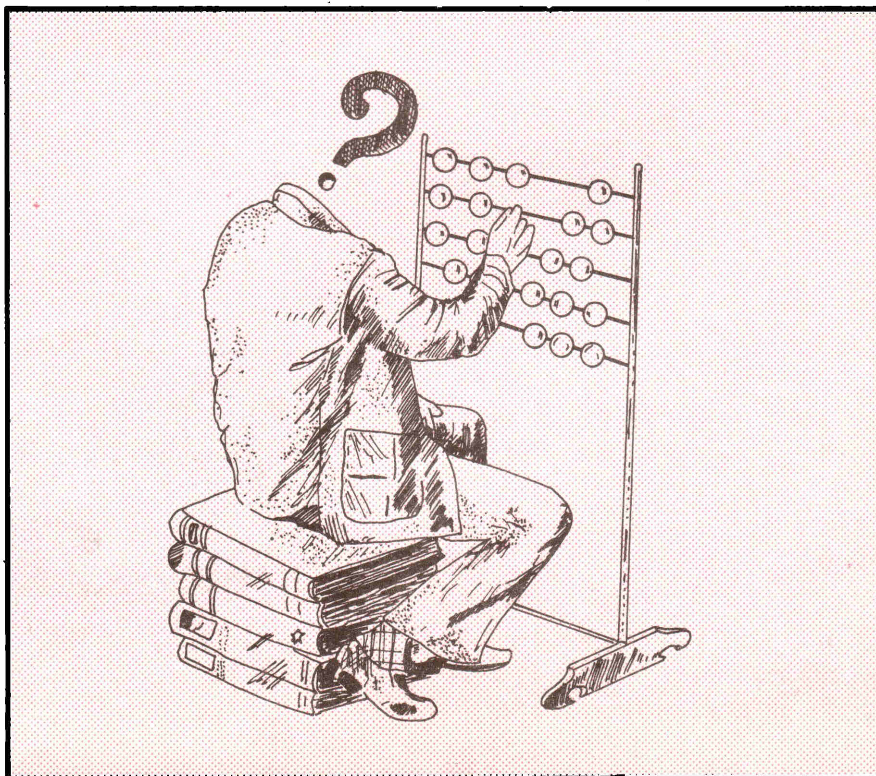
Per esempio:

40 DEF FN y(x) = x COS(x/2)

Quindi, si dà un CONTINUE, e vengono chiesti l'inizio (x1) e la fine (x4) dell'intervallo. Poniamo x1=0 e x4=PI; e l'intervallo minimo D (esempio: 0.0001).

Questo richiede una spiegazione: per trovare il massimo, cercato ottimizzando in qualche modo i calcoli (cioè senza farne un'infinità), un metodo abbastanza semplice è il seguente: dato l'intervallo da studiare (tra x1 e x4), si sceglie un piccolissimo intervallo, largo appunto D, al centro di x4-x1, avente per estremi x2 e x3.

Evidentemente avremo che:



x2 = x1 + (x4-x1) /2 - D/2
x3 = x1 + (x4-x1) /2 + D/2 = x2 +D

Si calcola la funzione in x2 e in x3; se risulta y2 maggiore di y3, il massimo che cerchiamo è compreso tra x1 e x3, e possiamo scartare subito tutto l'intervallo x3-x4 (viceversa se fosse stato y2 minore di y3).

Ora facciamo diventare x4 = x3 (o x1 = x2 nell'altro caso), scegliamo altri due punti: x2 e x3 al centro del nuovo intervallo e ripetiamo la procedura.

Come si vede, l'intervallo x4-x1 si restringe geometricamente: quella che si dice una convergenza rapida. Quando x4-x1 diventa minore di 3D, la ricerca ha termine. Il programma tabula tutti i valori di x e y calcolati, fino al massimo cercato. ■




```
   1 REM "maxfn"
   5 REM ↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑
  10 REM ↑ Ricerca del massimo ↑
  20 REM ↑↑↑ di una FUNZIONE ↑↑↑
  30 REM ↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑↑
  35 GO SUB 500
  40 DEF FN y(X)=x*COS x
  45 INPUT "margine sinistro (x1
) = ";x1
  50 PRINT
  60 INPUT "margine destro (x4)
= ";x4
  80 PRINT
  90 INPUT "intervallo minimo (d
) = ";d
 110 LET i=1
 130 REM  calcolo punti interni
 150 LET x2=x1+.5*(x4-x1-d)
 160 LET x3=x2+d
 170 LET y2=FN y(x2)
 180 LET y3=FN y(X3)
 190 PRINT
 200 PRINT "y1=";FN y(x1);TAB 16
;"x1=";x1,"y2=";y2;TAB 16;"x2=";
x2
 210 PRINT "y3=";y3;TAB 16;"x3="
;X3,"y4=";FN y(x4);TAB 16;"x4=";
x4
 220 IF y2<y3 THEN GO TO 300
 230 IF y2=y3 THEN GO TO 400
 250 REM interv.sinistro
 270 LET x4=x3
 280 GO TO 340
 300 REM interv.destro
 320 LET x1=x2
 340 REM test per uscita
 360 IF i=100 THEN GO TO 470
 370 LET i=i+1
 380 IF x4-x1>3*d THEN GO TO 130
 400 REM ↑↑ soluzione finale
 420 LET x5=.5*(x2+x3)
 430 PRINT
 440 PRINT "yMAX=";FN y(x5),"xMA
X=";x5
 370 LET i=i+1
 380 IF x4-x1>3*d THEN GO TO 130
 400 REM ↑↑ soluzione finale
 420 LET x5=.5*(x2+x3)
 430 PRINT
 440 PRINT "yMAX=";FN y(x5),"xMA
X=";x5
 450 STOP
 470 REM ↑↑ superam.iterazioni
 490 PRINT "Troppe iterazioni","
fine elaborazione": GO TO 450
 500 CLS : PRINT ,,"Scrivi la se
guente linea di       programma:",
,,," 40 DEF FN y(x)=<la funzion
e di"," x che vuoi stu"," diar
e>"
 510 PRINT ,,,,"... quindi premi
 CONT": STOP
 520 RETURN
1000 CLEAR
1010 SAVE "maxfn"
1020 STOP
```

Forse non tutti sanno che ...

ZX81

• **Alla locazione 16418** la variabile di sistema contiene il numero di linee della parte bassa dello schermo (normalmente 2).

Se in un programma volete usare tutto il video, potete farlo scrivendo zero in questa variabile. Attenzione però: gli interventi della tastiera ripristinano il valore originario, e anche successive PRINT non possono essere fatte, senza incappare nell'errore 5.

```
10 POKE 16418,0
20 FOR K=1 TO 24
30 PRINT TAB 15;K
40 NEXT K
```

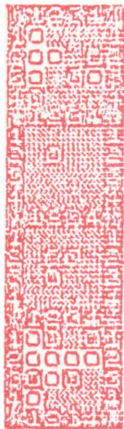
Tuttavia, si può conservare intatto (finchè il programma gira) quanto scritto nelle ultime due righe del video, riscrivendo POKE 16418,2.

Filetto.... suppergiù

UN classico, nel suo genere, questo filetto da giocare su quattro campi contemporaneamente. Il gioco è tanto conosciuto da non richiedere spiegazioni; le mosse vanno digitate come numeri di tre cifre, senza spazi o virgole. Il computer, in verità non sembra difendersi molto bene, e vi lascia vincere con discreta facilità: più che da avversario, si comporta da allenatore, spargendovi palline qua e là, senza una chiara strategia, nè un'apparente voglia di vincere. Quindi suggeriamo di giocare alla rovescia: cercate disperatamente di perdere.

Quanto al vero filetto, invece, chi ce ne manda uno efficiente, sviluppato ex-novo o partendo da questo? Condizione unica da rispettare è che si giochi simultaneamente su 4 tavolieri di 4x4 caselle; ricordate che sono validi anche i filetti diagonali che attraversano i 4 tavolieri.

Interessante, in questo listato, è l'abbondante uso che viene fatto degli operatori logici, che permettono una notevole compattezza nell'esame delle condizioni multiple.



```
SCRIVI LA MOSSA
IN QUESTO ORDINE:
1. COORDINATA X
2. COORDINATA Y
3. CAMPO

1 3 4

HAI VINTO
```

```
   1 REM "TRIFIL"
   2 REM **** SUPERFILETTO ****
   3 REM $ FRIEDRICH KREH $
   4 REM $ ZX-USER CLUB/RFT $
   5 CLS
  10 REM CAMPO DI GIOCO
  20 FOR I=1 TO 4
  30 PRINT "[illegible]"
  40 PRINT "[illegible]"
  50 PRINT "[illegible]"
  60 PRINT "[illegible]"
  70 PRINT "[illegible]"
  80 NEXT I
  90 PRINT "[illegible]"
 100 PRINT AT 0,15;"SCRIVI LA MO
SSA";TAB 15;"IN QUESTO ORDINE:"
 110 PRINT TAB 15;"1. COORDINATA
 X";TAB 15;"2. COORDINATA Y";TAB
 15;"3. CAMPO"
 120 REM VARIABILI
 130 DIM A(4,4,4,7)
 140 LET P=PEEK 16396+256*PEEK 1
6397
 150 LET P=P+1-33*5
 160 LET G=5
 200 REM MOSSA DEL GIOCATORE
 210 INPUT A$
 220 IF LEN A$<>3 THEN GOTO 210
 230 IF A$(1)<"1" OR A$(2)<"1" O
R A$(3)<"1" OR A$(1)>"4" OR A$(2
)>"4" OR A$(3)>"4" THEN GOTO 210
 240 LET X=VAL A$(1)
 250 LET Y=VAL A$(2)
 260 LET Z=VAL A$(3)
 270 LET F=PEEK (P+X+33*Y+33*5*Z
)
 280 IF F<>8 AND F<>136 THEN GOT
O 210
 290 POKE (P+X+33*Y+33*5*Z),52
 300 LET E=-1
 310 GOSUB 1000
 320 REM MOSSA DEL COMPUTER
 335 LET M=0
 340 FOR I=1 TO 4
 345 LET C=I=2 OR I=3
 350 FOR J=1 TO 4
 355 LET D=J=2 OR J=3
 360 FOR K=1 TO 4
 362 LET O=K=2 OR K=3
 364 LET F=PEEK (P+I+33*J+33*5*K
)
 365 IF F<>8 AND F<>136 THEN GOT
O 500
 368 DIM B(10)
 370 FOR L=1 TO 7
 380 LET B(A(I,J,K,L)+5)=B(A(I,J
,K,L)+5)+1
 390 NEXT L
 395 LET N=.5
 400 IF B(4) THEN LET N=1
 410 IF B(6) THEN LET N=2.7
 420 IF B(3) THEN LET N=3
 425 IF B(3) AND B(4)=1 THEN LET
N=4
 430 IF B(3)>1 THEN LET N=5
 440 IF B(7)>1 THEN LET N=6
 450 IF B(2) THEN LET N=7
```

```
 460 IF B(6) THEN GOTO 500
 465 LET N=N+(C+D+Q)/5
 470 IF N<M THEN GOTO 500
 460 IF N=M THEN IF RND>.5 THEN
GOTO 500
 490 LET M=N
 492 LET X=I
 494 LET Y=J
 496 LET Z=K
 510 NEXT K
 520 NEXT J
 530 NEXT I
 540 LET E=1
 550 PRINT AT 5,15;X;"  ";Y;"  ";Z
 560 POKE P+X+33*Y+33*5*Z,160
 570 SLOW
 560 GOSUB 1000
 590 GOTO 210
 600 REM VINCI
 610 POKE P+I+33*J+33*5*K,160
 620 GOTO 920
 900 REM PERDI
 910 PRINT AT 15,15;"HAI VINTO"
 920 PAUSE 4E4
 925 SLOW
1000 REM FILETTO
1010 IF A(X,Y,Z,1)=G THEN GOTO 1
070
1020 LET I=1
1030 LET X$="J"
1040 LET Y$="Y"
1050 LET Z$="Z"
1060 GOSUB 1500
1070 IF A(X,Y,Z,2)=G THEN GOTO 1
130
1060 LET I=2
1090 LET X$="X"
1100 LET Y$="J"
1110 LET Z$="Z"
1120 GOSUB 1500
1130 IF A(X,Y,Z,3)=G THEN GOTO 1
190
1140 LET I=3
1150 LET X$="X"
1160 LET Y$="Y"
1170 LET Z$="J"
1160 GOSUB 1500
1190 IF X<>Y AND X<>5-Y OR A(X,Y
,Z,4)=G THEN GOTO 1250
1200 LET I=4
1210 LET X$="J"
1220 LET Y$=("J" AND X=Y)+("5-J"
 AND X=5-Y)
1230 LET Z$="Z"
1240 GOSUB 1500
1250 IF E=-1 THEN FAST
1255 IF A(X,Y,Z,5)=G OR X<>Z AND
 X<>5-Z THEN GOTO 1310
1260 LET I=5
1270 LET X$="J"
1230 LET Y$="Y"
1290 LET Z$=("J" AND X=Z)+("5-J"
 AND X=5-Z)
1300 GOSUB 1500
1310 IF A(X,Y,Z,6)=G OR Y<>Z AND
 Y<>5-Z THEN GOTO 1370
1320 LET I=6
1330 LET X$="X"
1340 LET Y$="J"
1350 LET Z$=("J" AND Y=Z)+("5-J"
 AND Y=5-Z)
1360 GOSUB 1500
1370 IF A(X,Y,Z,7)=G OR X<>Y AND
 X<>5-Y OR Y<>Z AND Y<>5-Z THEN
RETURN
1380 LET I=7
1390 LET X$="J"
1400 LET Y$=("J" AND Y=X)+("5-J"
 AND X=5-Y)
1410 LET Z$=("J" AND Z=X)+("5-J"
 AND X=5-Z)
1500 IF SGN A(X,Y,Z,I)=-E THEN L
ET A(X,Y,Z,I)=G
1510 IF A(X,Y,Z,I)<>G THEN LET A
(X,Y,Z,I)=A(X,Y,Z,I)+E
1516 IF A(X,Y,Z,I)=-4 THEN GOTO
900
1520 FOR J=1 TO 4
1530 LET A(VAL X$,VAL Y$,VAL Z$,
I)=A(X,Y,Z,I)
1540 NEXT J
1550 RETURN
9000 SAVE "TRIFI■"
9001 RAND
9002 RUN
```

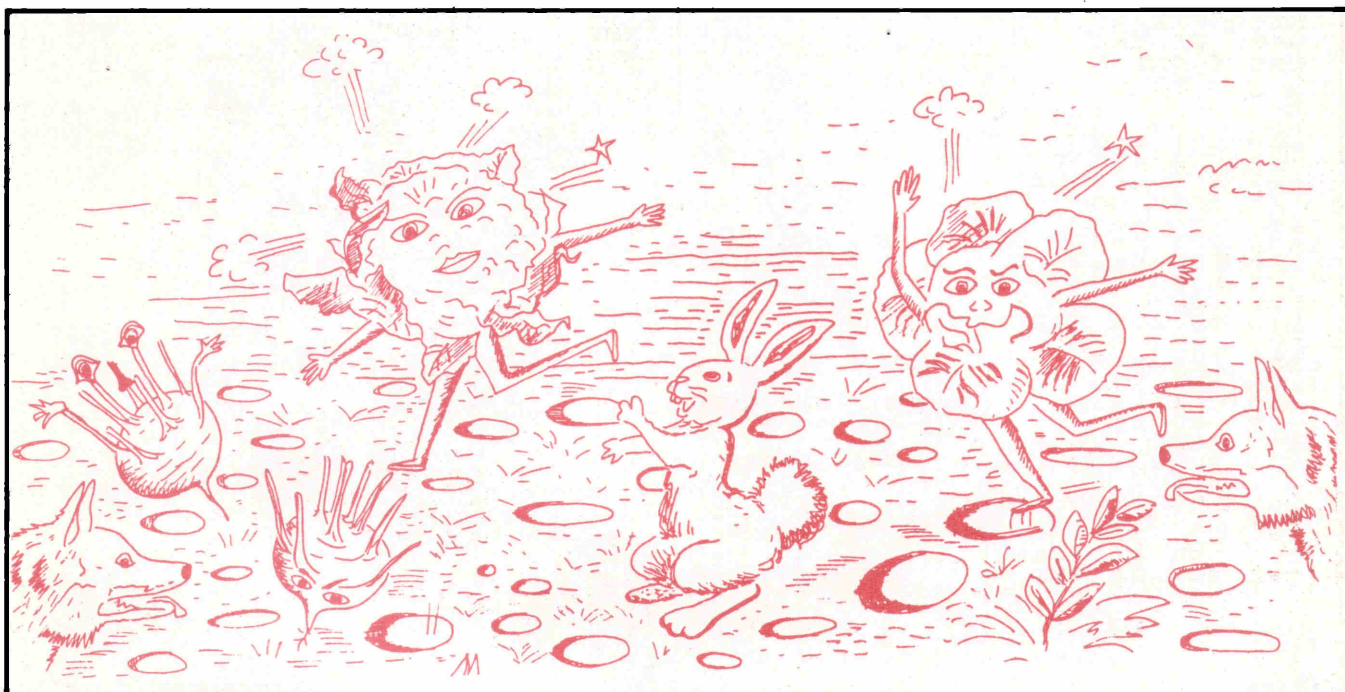
Caratteri ingranditi

ZX81

```
  10 REM STAMPA CARATTERI
         INGRANDITI
1000 FOR X=16476 TO 16507
1010 POKE X,0
1020 NEXT X
1030 POKE 16508,118
1040 LPRINT "(STRINGA)"
1050 RETURN
```

QUESTA ROUTINE SERVE PER PRODUR-RE ALLA STAMPANTE SCRITTE INGRANDITE.
OPERA "ALLARGANDO" IL BUFFER DI STAMPA, SPOSTANDO IN AVANTI IL NEWLINE IN UNA AREA DI LAVORO NORMALMENTE LIBERA (MA NON SEMPRE: A VOLTE CON VARIABILI $ INDICIATE NON FUNZIONA).

Un coniglio nei PASTICCI



UN campo di cavoli, che però è una specie di groviera, con i cavoli che vanno e vengono, spuntano e spariscono. Buchi trabocchetto, strane volpi e una specie di falciatrice. Povero coniglio!

I tasti per farlo muovere sono, come al solito, 5-6-7-8. Parte in basso a destra, e cerca di mangiare più cavoli possibili; oltre a questo, vorrebbe visitare i suoi colleghi conigli sparsi qua e là, cercando di restare incolume.

Non sarà un gioco eccezionale, ma è dignitoso. Tutti i caratteri alfabetici, racchiusi tra virgolette dalla linea 100 in poi, sono da intendersi scritti in modo grafico, escluse le parole con un significato logico.

Queste lettere sono:

num. linea	carattere
100	prima f, poi e
120	b
160	a
190	f
200/230	h
235	f
240	g
250	d
310	h,h
320	g,g
330	d,d
8505 e 8520	32 h ciasc.
8525	h, 30 f, h



```
  10 GO SUB 9000: REM sel grafic
o
  20 LET h=0: BORDER 8: PAPER 4:
 CLS
  30 CLS : PRINT AT 0,5;"conigli
o!!": GO SUB 8500: REM prelimina
ri
  40 LET s=0: PRINT #0;"velocita
"(1=veloce; 9=lento)": PAUSE 0:
LET k=CODE INKEY$-48: IF NOT k O
R k>9 THEN INPUT "
                   ": GO TO 40
  50 INPUT "
        "


  60 LET v=5: LET w=5
  70 FOR n=1 TO 5: PRINT AT n,0;
 PAPER 5,,: NEXT n
  75 PRINT AT 2,15; PAPER 8;"rec
ord ";h
  80 LET x=20: LET y=29
 100 FOR j=29 TO 1 STEP -1: PRIN
T AT 5,j; PAPER 5; INK 4;" ⁎";AT
 5,j; INK 0;"⁎"
 110 GO SUB 200: REM sposta il c
oniglio
 120 PRINT AT x,y; PAPER 8; INK
7;"≟"
 125 LET s=s+1: PRINT PAPER 5;AT
 2,0;"punti ";s
 130 BEEP .0005,50
 135 IF x=v AND y=w THEN GO TO 7

000
 140 GO SUB 300: REM alter maze
 145 LET k=k-.001
 150 IF x=5 AND y=j THEN GO TO 7


000
 160 GO SUB 400: PRINT AT v,w; P
APER 8; INK 6;"≛"
 170 PRINT AT x,y; PAPER 8;" "
 180 NEXT j
 190 PRINT AT 5,1; PAPER 5; INK
4;"⁎": GO TO 100
 200 IF INKEY$="5" AND m$(x,y-1)
<>"+" THEN LET y=y-1
 210 IF INKEY$="8" AND m$(x,y+1)
<>"+" THEN LET y=y+1
 220 IF INKEY$="7" AND m$(x-1,y)
<>"+" THEN LET x=x-1
 230 IF INKEY$="6" AND m$(x+1,y+
1)<>"+" THEN LET x=x+1
 235 IF m$(x,y)="⁎" THEN LET s=s
+10
 240 IF m$(x,y)="≙" THEN GO TO 7

000
 250 IF m$(x,y)="≛" THEN LET s=s
+100: LET m$(x,y)=" ": BEEP .5,0
: BEEP .0125,30
 299 RETURN
 300 IF RND<.9 THEN RETURN
 305 LET a=INT (RND*14)+6: LET b
=INT (RND*27)+2
 307 IF RND<.5 THEN GO TO 320
 310 LET m$(a,b)="+": PRINT AT a
,b;"+"
 320 LET m$(a,b)="≙": PRINT AT a
,b;"≙"
 325 IF RND<.7 THEN RETURN
 330 LET m$(a,b)="≛": PRINT AT a
,b; INK RND*3;"≛"
 399 RETURN
 400 PRINT PAPER 8;AT v,w;m$(v,w
)
 410 IF RND<(k/12) THEN RETURN
 420 IF x<v THEN LET v=v-1
 430 IF x>v THEN LET v=v+1
 440 IF y>w THEN LET w=w+1
 450 IF y<w THEN LET w=w-1
 460 IF x=v AND y=w THEN GO TO 7

000
 470 RETURN
7000 FOR l=15 TO 1 STEP -1: BEEP
 .01,1: NEXT l: CLS : PRINT "PUN
TI: ";s: IF s>h THEN PRINT "part
ita record": LET h=s
7010 PRINT "Un tasto qualsiasi p
er giocare  ancora": PAUSE 0

7015 CLS
7030 GO TO 30
8500 DIM m$(21,31)
8505 LET m$(21)="+++++++++++++++
++++++++++++++++"
8510 FOR n=5 TO 21: LET m$(n,1)=
"+": LET m$(n,31)="+": NEXT n
8520 LET m$(4)="+++++++++++++++
++++++++++++++++"
8525 LET m$(5)="+++++++++++++++
++++++++++++++++"
8530 FOR n=1 TO 100: LET m$(INT
(RND*14)+6,INT (RND*30)+1)="+":
NEXT n
8540 FOR n=5 TO 21: PRINT AT n,0
;" ";m$(n): NEXT n
8550 RETURN

9000 FOR g=1 TO 7: READ a$: FOR
n=0 TO 7: READ a: POKE USR a$+n,
a: NEXT n: NEXT g
9010 DATA "A",0,BIN 01000010,BIN
 01100110,BIN 01111110,BIN 01011
010,BIN 01111110,BIN 01000010,BI
N 01111110
9020 DATA "B",0,BIN 00100100,BIN
 00100100,BIN 00111100,BIN 11011
011,BIN 01111110,BIN 11000011,BI
N 00111100


9030 DATA "D",BIN 00100100,BIN 0
0100100,BIN 00111100,BIN 0101101
0,BIN 00111100,BIN 01111110,BIN
01111110,BIN 00111100
9040 DATA "E",BIN 00001000,BIN 0
0000100,BIN 00001010,BIN 0001000
1,BIN 01110000,BIN 11110000,BIN
11110000,BIN 01100000
9050 DATA "F",0,0,0,0,BIN 000110
00,BIN 00111100,BIN 00111100,BIN
 00011000

9060 DATA "G",BIN 00111100,BIN 0
1111110,255,255,255,255,BIN 0111
1110,BIN 00111100
9070 DATA "H",0,0,BIN 00011000,B
IN 00111100,BIN 00111100,BIN 000
11000,0,0

9499 RETURN
9900 CLEAR
9910 SAVE "ords"
9920 STOP
```

“Solitaire”

“Un gioco lineare (?!), semplice che più non si può...
...Meditate, gente, meditate”



UNA linea, e nulla più. Una linea orizzontale; lenta, per giunta. Con il tasto “5” la disegnate, con il tasto “8” la cancellate. Oppure con il joystick, rispettivamente a sinistra e a destra. Non ci sono regole. Scrivete la linea, cancellatela. Pensateci su. Provate a fare una linea tratteggiata (non è facile come sembra). Rilassatevi. Consigliato come esercizio di training autogeno.

```
10 REM **** ROUT/
15 LET MAT=1
20 LET X=0
30 IF INKEY$<>"" THEN GOTO 50
31 IF MAT=1 THEN UNPLOT X,10
32 PLOT X,10
33 IF MAT=0 THEN UNPLOT X,10
40 GOTO 100
50 IF INKEY$<>"8" THEN GOTO 90
60 LET X=X+1
65 IF X>63 THEN LET X=0
70 IF MAT=1 THEN PLOT X,10
72 IF MAT=1 THEN UNPLOT X,10
80 IF INKEY$="8" THEN GOTO 60
90 IF INKEY$<>"5" THEN GOTO 12
1
 100 LET X=X-1
 105 IF X<0 THEN LET X=63
 110 IF MAT=1 THEN PLOT X,10
 112 IF MAT=0 THEN UNPLOT X,10
 120 IF INKEY$="5" THEN GOTO 100
 121 IF INKEY$<>"8" THEN GOTO 13
0
 122 LET MAT=MAT+1
 123 IF MAT>=2 THEN LET MAT=0
 124 IF INKEY$="" THEN GOTO 130
 125 GOTO 124
 130 GOTO 30
```

Piccoli universi

"Ovvero, fatevi il vostro universo"

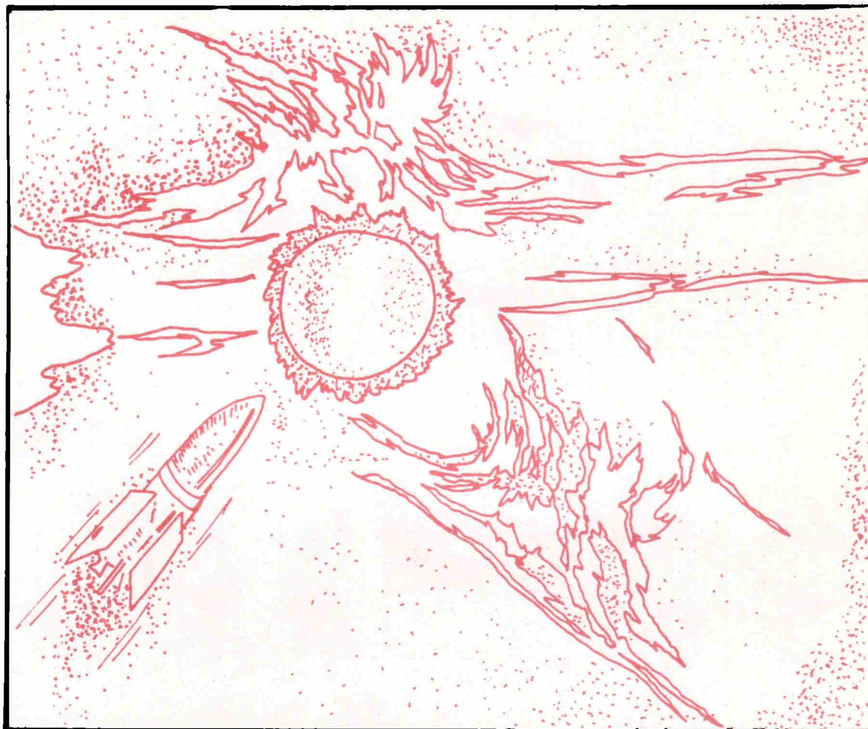
UN sole al centro, che produce un campo gravitazionale un po' particolare, attirando i corpi celesti a 45° (escluse la linea e la colonna occupate dal sole stesso). La distanza di un pianeta dal sole è data dalla somma dei valori assoluti delle coordinate X e Y. L'attrazione (un po' di rispetto almeno per Newton) è inversamente proporzionale al quadrato della distanza:

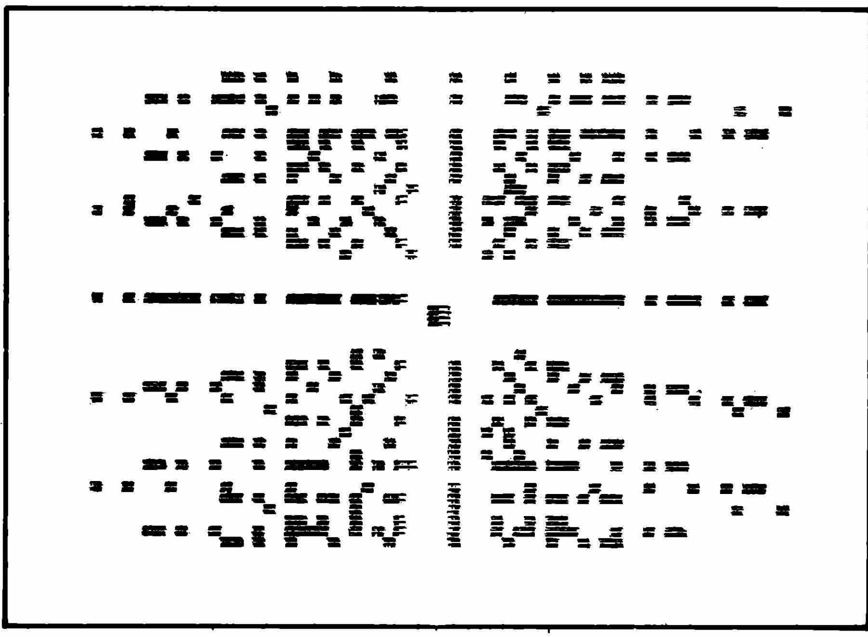
$$G=1+INT\frac{29}{[1+ABS(X)+ABS(Y)]^2}$$

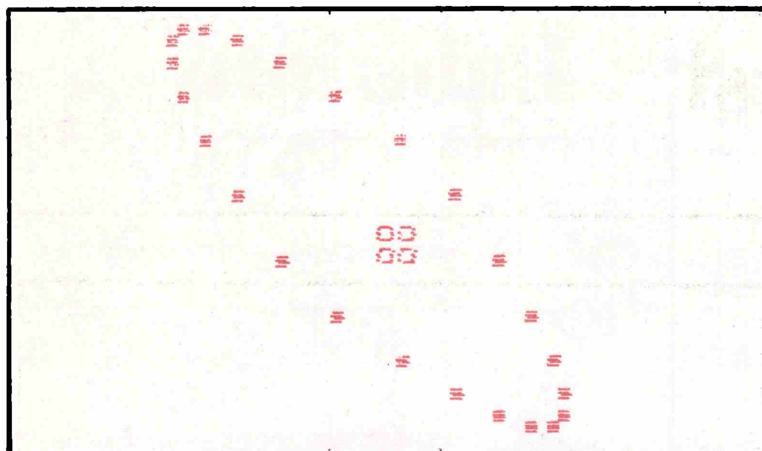
E siccome siete voi il Creatore, non vi dovete giustificare con nessuno quando vorrete cambiare qualcosa. Dovete fornire una posizione di partenza del corpo celeste, dando le coordinate X e Y qualsiasi (ma piccoli numeri, oppure andate fuori schermo), e le due componenti della velocità. Quindi state a guardare.

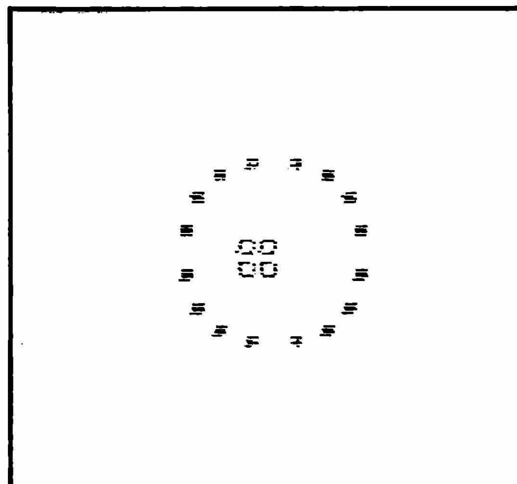
Il programma fa una pausa ogni 50 cicli, ma se volete vedere di più, eliminate il FAST. Al millesimo ciclo viene eseguito automaticamente un COPY dello schermo.

A volte, si ottengono disegni casuali, altre perfettamente simmetrici, secondo le coordinate fornite. Alcuni esempi sono riprodotti. ▪

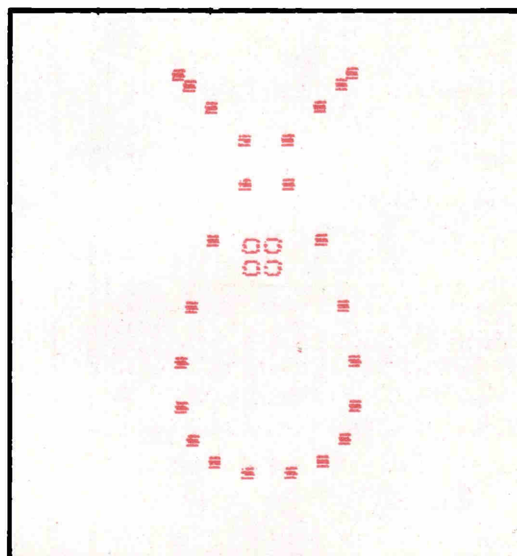


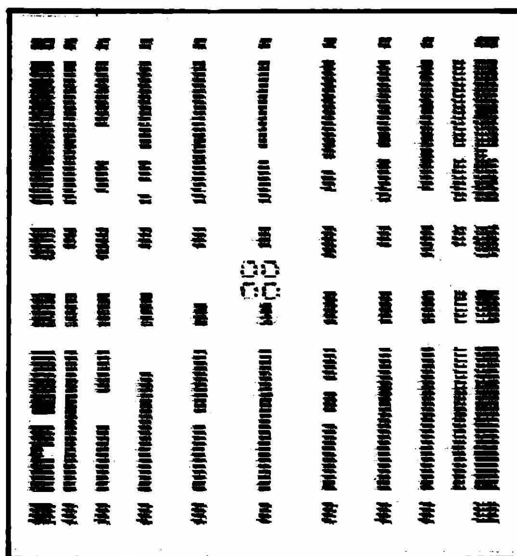






```
   2 REM ----------------------
   3 REM -- PERSONAL UNIVERSE --
   4 REM  J.S.BEUGUE/TELESOFT 83
   5 REM ----------------------
   6 LET K=0
   7 FAST
   8 CLS
  10 PRINT "IMPOSTA IN QUESTO OR
DINE."
  15 PRINT "COORD. X",,"COORD. Y
",,"COMPONENTI DELLA VELOCITA","
 (VX) E (VY)"
  20 INPUT X
  25 INPUT Y
  30 INPUT VX
  35 INPUT VY
  40 CLS
  45 POKE 16418,0
  50 PRINT AT 22,0;"X=";X;" Y=";
Y;" VX=";VX;" VY=";VY
  55 POKE 16418,1
  60 LET G=1+INT (29/((1+ABS X+A
BS Y)**2))
  70 LET GX=G*-SGN X
  80 LET GY=G*-SGN Y
  90 LET VX=VX+GX
 100 LET VY=VY+GY
 110 LET X=X+VX
 120 LET Y=Y+VY
 130 GOSUB 500
 135 LET K=K+1
 140 IF INT (K/50)=K/50 THEN PAU
SE 100
 150 POKE 16437,255
 155 IF K<1000 THEN GOTO 60
 160 COPY
 165 STOP
 500 LET A=31+X
 510 IF A<0 OR A>63 THEN GOTO 56
0
 520 LET B=21+Y
 530 IF B<0 OR B>43 THEN GOTO 56
0
 540 UNPLOT A,B
 550 PLOT A,B
 560 PRINT AT 10,15;"OO"
 570 PRINT AT 11,15;"OO"
 580 RETURN
9000 CLEAR
9010 SAVE "DRAW/█"
9020 RUN
```

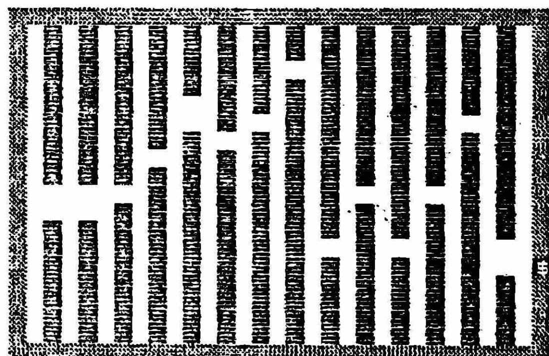




Labirinto

Come uscire da questo semplice labirinto?

```
   1 REM "LABIR"
   2 REM ▒▒▒▒▒▒▒▒▒▒▒LABIRINTO▒▒▒
   3 REM ▒▒▒▒▒▒▒▒▒▒▒PAZZO▒▒▒▒▒▒▒
   4 REM TIM HARTNELL/GB********
   5 LET U=0
  10 FAST
  15 FOR B=2 TO 28 STEP 2
  20 FOR A=0 TO 19
  30 PRINT AT A,B;"▒"
  40 NEXT A
  45 PRINT AT RND*14+3,B;" "
  50 NEXT B
  60 FOR A=0 TO 30
  70 PRINT AT 0,A;"▒";AT 19,A;"▒
"
  80 NEXT A
  90 FOR B=1 TO 18
 100 PRINT AT B,0;"▒";AT B,30;"▒
"
 110 NEXT B
 120 GOSUB 9000
 125 GOTO 167
 150 PRINT AT A,B;"▒"
 155 LET Z=Z-673
 160 PRINT AT 20,7;Z;"        "
 165 RETURN
 167 LET Z$=INKEY$
 170 IF Z$="" THEN LET Z$=A$
 180 LET Z=Z-50
 200 LET Y=A
 210 LET X=B
 220 LET A=A+(Z$="A")-(Z$="D")
 230 LET B=B+(Z$="S")
 232 PRINT AT Y,X;" "
 235 IF PEEK (PEEK 16396+256*PEE
K 16397+33*A+B+1)=128 THEN LET Z
=Z-673
 244 PRINT AT A,B;"$"
 255 IF A>18 OR A<2 OR B<1 THEN
LET Z=INT (Z/3)
 260 IF A>18 OR A<2 OR B<1 OR B>
29 THEN GOTO 510
 490 LET A$=Z$
 500 GOTO 167
 510 PRINT AT 20,0;"FINE DELLA M
ANCHE. PUNTI: ";Z
 520 IF Z>U THEN LET U=Z
 530 FOR G=1 TO 8
 540 PRINT AT 21,3;" RECORD : ";
U
 545 PRINT AT A,B;"▒";AT A,B;"$"
;AT A,B;"▒"
 550 PRINT AT 21,3;"RECORD"
 560 NEXT G
 570 CLS
 580 GOTO 10
 950 RETURN
9000 LET A=10
9010 LET B=1
9020 LET Z=20000
9030 LET Y=A
9040 LET X=B
9050 LET A$="Z"
9400 SLOW
9500 RETURN
```



```
FINE DELLA MANCHE. PUNTI: 13608
      RECORD : 13608
```



Lo ZX81 disegna un labirinto, formato da 14 porte sparse su altrettanti muri.

Voi siete il "$" che appare alla sinistra e vi muovete con i tasti A, D, S, rispettivamente per basso, alto, destra (o qualsiasi altro tasto preferiate: vengono definiti alle linee 220/230).

Dovete arrivare al lato opposto dello schermo nel minor tempo possibile, e senza abbattere troppo muro. Il punteggio (che appare solo alla fine) parte da 20.000 e decresce costantemente, crollando drasticamente ogni volta che mancate una porta e sbattete in un muro. Se si arriva al lato superiore o inferiore, la manche si interrompe; il programma conserva il miglior punteggio realizzato. Dovreste superare i 13.500 punti.

Analisi di un testo poetico

"Un esperimento d'uso dello ZX81 come strumento didattico: l'analisi fonetica di un testo. Perché in una poesia allegra ci sono più "i" che "u"?

di C. Porcelli



Come vedete, c'è anche chi usa lo ZX81, che molti ritengono solo un simpatico giocattolo, per cose molto serie.

Nel programma in questione, andrebbe un po' migliorato il sistema di inserimento del testo, laborioso e poco pratico, quando si deve correggere (si deve rifare tutto). Durante le elaborazioni ci si trova troppo spesso in errore 5 (schermo pieno).

Insieme al listato, trovate un esempio di uso della prima opzione *(Omotonie)*, impiegata su qualche verso del buon vecchio Leopardi.

QUESTO programma, realizzato per lo ZX81 con espansione di memoria di 16k, è in grado di aiutare chi desidera effettuare un esame fonologico di un testo poetico; i quattro lavori previsti riguardano le *omotonie, le rime, i palindromi, le allitterazioni.*

Prima di illustrarli dettagliatamente, occorre rilevare che i programmi come questo sono utili solo nelle lingue, in cui la grafia rappresenta abbastanza regolarmente la pronuncia; in caso contrario, si deve far ricorso ai simboli dell'alfabeto fonetico e lavorare esclusivamente sulla trascrizione della pronuncia. Ma sappiamo che, anche in italiano, vi sono delle incongruenze e irregolarità. Se ci si basa sulla grafia corrente, il computer riconosce come sequenze identiche "SC" di "liscia" e "SC" di "lisca", che inve-

ce rappresentano suoni diversi. Lo stesso vale per GL in "egli" e "inglese", C in "cane" e in "cena", G in "gola" e in "gelo", I in "ciao" e in "via", ecc.

L'utente deve scegliere se lasciare il testo poetico nella grafia corrente, per poi trascurare le eventuali incongruenze e imprecisioni, ovvero digitare il testo usando, ad esempio, le lettere J, K, W, X, Y per discriminare preventivamente le possibili ambiguità. Così facendo, "lisca" potrebbe diventare LISKA e "liscia" LIXA: verrebbero poi analizzate le trascrizioni e interpretati i risultati. Sulla base delle esperienze effettuate, la prima soluzione è normalmente preferibile, e perciò non è stata inserita nel programma la possibilità di far compiere al computer stesso il lavoro di trascrizione e riconversione.

I quattro sottoprogrammi

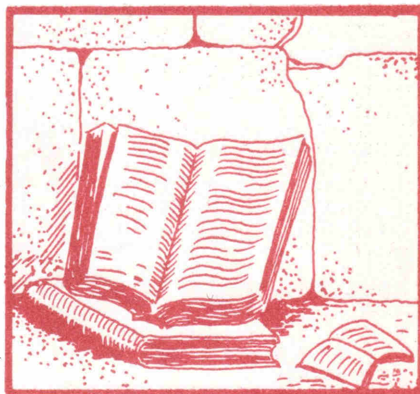
1. Le omotonie sono le ricorrenze dello stesso suono vocalico. Vi sono, come è noto, vocali più o meno aperte o chiuse, e vocali più o meno gravi o acute. La presenza, a volte, insistente, di un certo suono in un verso o in una strofa, riesce a dare una tipica "coloritura" al brano poetico, e molti autori usano sapientemente i dosaggi dei vari suoni, i passaggi da una vocale dominante all'altra, l'alternarsi contrastante di toni "limpidi e cristallini" o "lugubri e cupi". Il primo sottoprogramma evidenzia, per mezzo di caratteri video inversi, la presenza e la distribuzione delle cinque vocali, una alla volta.

2. Il termine rime, non ha bisogno di chiarimenti; il sottoprogramma, però, lo interpreta in maniera estensiva, evidenziando tutte le sequenze ricorrenti di almeno due lettere, in qualsiasi posizione: finale di verso o di parola, iniziale o intermedia.

3. I palindromi sono, in enigmistica, quelle parole o frasi che possono essere lette indifferentemente iniziando dalla sinistra o dalla destra: ANILINA, ASIL GELOSO SOL EGLI SA. Con riferimento alla poesia si parla anche di "rime a specchio", nelle quali una sequenza di suoni viene riproposta in ordine inverso: "teneramente sul mare". Come nel caso della rima, questa forma di "eco" serve a legare tra loro le parole e a creare effetti sonori particolari.

4. Le allitterazioni sono molto importanti non solo nel linguaggio letterario, ma anche in quello di ogni giorno. Si pensi ad espressioni comuni come "sano e salvo", "vivo e vegeto", "tagliamo la testa al toro", e così via, caratterizzate dalla ripetizione della consonante iniziale. Molti slogan, pubblicitari e non, sono basati sull'allitterazione, e nelle lingue del gruppo germanico essa è spesso sentita e usata ancor più dalla rima.

Questa sezione è suddivisa in due sottoprogrammi. Il primo analizza se in posizione iniziale vi siano sequenze identiche di almeno due lettere (i cosiddetti nessi consonantici, come in "primo programma"); questa opzione è superflua, se il testo è già stato analizzato col sottoprogramma Rima. Il secondo sottoprogramma, invece, evidenzia le allitterazioni limitatamente alla sola lettera iniziale.



Il testo in input, va digitato senza la punteggiatura che, oltre ad essere del tutto irrilevante ai fini dell'esame fonologico, crea problemi in alcuni sottoprogrammi. La lunghezza massima del testo (spazi compresi) deve essere tale da consentire la visualizzazione dell'intero brano sullo schermo; in caso contrario, il comando CONT fa ritornare sempre sulla stessa linea. Quasi tutti i sonetti e molte poesie di quattro quartine rientrano in questo limite; i testi più lunghi devono invece essere segmentati e analizzati a più riprese. I versi vanno scritti l'uno di seguito all'altro, eventualmente divisi mediante "/"; un endecasillabo, infatti, occupa di solito più di 32 caratteri, e andare a capo, dopo ogni verso, significa poterne esaminare al massimo undici per volta. Inoltre, la presenza di tutti gli spazi aggiuntivi rallenta ulteriormente l'esecuzione dei lavoro.

```
   3 REM ** ANALISI DI POESIE **
   5 REM **    DI G. PORCELLI   **
   7 REM **          1983        **
  10 GOTO 200
  30 PRINT
  32 PRINT "********************
***********"
  34 PRINT
  36 RETURN
 200 REM ** INPUT **
 210 GOSUB 30
 230 PRINT "BATTI IL TESTO DA AN
ALIZZARE","PREMI <NEWLINE> SOLO
ALLA FINE."
 240 GOSUB 30
 250 INPUT A$
 260 CLS
 270 GOSUB 30
 290 PRINT A$
 300 GOSUB 30
 310 PRINT "BATTI ""L"" SE VA BE
NE,",""""K"" PER CORREGGERE"
 315 SLOW
 320 IF INKEY$="" THEN GOTO 320
 325 CLS
 330 IF INKEY$="K" THEN RUN
 335 LET B$=A$
 337 LET C$=A$
 340 REM **SCELTA LAVORO**
 345 GOSUB 30
 350 PRINT "ORA SCEGLI:"
 355 PRINT
 360 PRINT TAB 7;"■ 1    OMOTONIE"
 365 PRINT
 370 PRINT TAB 7;"■ 2    RIME"
 375 PRINT
 380 PRINT TAB 7;"■ 3    PALINDROM
I"
 385 PRINT
 390 PRINT TAB 7;"■ 4    ALLITTERA
ZIONI"
 400 GOSUB 30
 410 SLOW
 420 IF INKEY$="" THEN GOTO 420
 425 LET F1=VAL INKEY$
 430 FAST
 440 CLS
 450 GOTO F1*1000
1000 REM **OMOTONIE**
1002 LET A$=C$
1005 LET B$=A$
1010 LET D$="AEIOU"
1020 FOR G=1 TO 5
1040 LET R=0
1050 FOR F=1 TO LEN A$
1055 IF A$(F)=D$(G) THEN LET R=1
1060 IF A$(F)=D$(G) THEN LET A$(
F)=CHR$ (CODE D$(G)+128)
1070 NEXT F
1075 IF NOT R THEN GOTO 1110
1080 PRINT D$(G);":  ";A$
1090 GOSUB 30
1100 LET A$=B$
1110 NEXT G
1200 GOTO 8000
2000 REM **RIME**
2010 LET A$="    "+C$
2015 LET A=LEN A$
2020 FOR N=A TO 3 STEP -1
2030 FOR M=3 TO N-1
2040 IF CODE A$(N)<28 THEN GOTO
2220
2050 IF A$(N)<>A$(M) THEN GOTO 2
210
2060 LET B$=A$
2070 LET X=N
2080 LET Y=M
2090 LET B$(Y)=CHR$ (CODE B$(Y)+
128)
2100 LET B$(X)=CHR$ (CODE B$(X)+
128)
2110 IF A$(Y-1)=" " OR A$(X-1)<>
A$(Y-1) THEN GOTO 2200
2120 LET X=X-1
2130 LET Y=Y-1
2140 GOTO 2090
2200 IF X<N THEN PRINT B$(4 TO )
2205 IF X<N THEN PRINT
2210 NEXT M
2220 NEXT N
2300 GOTO 8000
3000 REM **PALINDROMI**
3005 LET A$=" "+C$
3010 LET B$=A$
3020 LET A=LEN A$
3030 FOR F=1 TO A
3040 FOR G=F+1 TO A
3060 IF A$(F)=A$(G) AND CODE A$(
F)>27 THEN GOTO 3200
3070 LET A$=B$
3100 NEXT G
3110 NEXT F
3120 GOTO 8000
3200 LET R=0
3210 LET S=CODE A$(F-R)
3220 IF S<28 THEN GOTO 3300
3230 LET S=S+128
3240 LET A$(F-R)=CHR$ S
3250 LET A$(G+R)=CHR$ S
3260 LET R=R+1
3270 IF F-R<=1 OR G+R>A THEN GOT
O 3300
3280 IF A$(F-R)=A$(G+R) THEN GOT
O 3210
3300 IF R>1 THEN PRINT A$
3305 IF R>1 THEN PRINT
3310 GOTO 3070
4000 REM **ALLITTERAZIONI**
4005 LET A$=" "+C$+" "
4010 LET B$=A$
4020 LET A=LEN A$
4030 GOSUB 30
4032 PRINT "SOTTOPROGRAMMI:"
4034 PRINT
4036 PRINT
4040 PRINT """1"" ANCHE NESSI IN
IZIALI","      DUE PER VOLTA"
4042 PRINT
4044 PRINT """2"" SOLO LE LETTER
E INIZIALI,      SIMULTANEAMENT
E"
4046 GOSUB 30
4048 SLOW
4050 IF INKEY$="" THEN GOTO 4050
4060 FAST
4065 CLS
4070 IF INKEY$="2" THEN GOTO 440
0
4200 FOR F=1 TO A
4210 IF A$(F)<>" " THEN GOTO 430
0
4230 FOR G=F+1 TO A
4240 IF A$(G)<>" " THEN GOTO 429
0
4245 LET R=1
4247 IF G+R>=A THEN GOTO 4300
4250 IF CODE A$(F+R)<38 OR A$(F+
R)<>A$(G+R) THEN GOTO 4280
4255 LET A$(F+R)=CHR$ (CODE A$(F
+R)+128)
4257 LET A$(G+R)=CHR$ (CODE A$(G
+R)+128)
4260 LET R=R+1
4270 GOTO 4250
4280 IF R=1 THEN GOTO 4290
4282 PRINT A$
4284 PRINT
4286 LET A$=B$
4290 NEXT G
4300 NEXT F
4310 GOTO 8000
4400 FOR F=38 TO 63
```

```
4420 LET R=0
4430 FOR G=1 TO A-2
4440 IF A$(G)<>" " THEN GOTO 4470
4445 IF CODE A$(G+1)=F THEN LET R=R+1
4450 IF CODE A$(G+1)=F THEN LET A$(G+1)=CHR$ (F+128)
4470 NEXT G
4480 IF R<=1 THEN GOTO 4500
4490 PRINT A$
4495 PRINT
4500 LET A$=B$
4510 NEXT F
```

```
********************************
   SILVIA RIMEMBRI ANCORA/QUEL T
EMPO DELLA TUA VITA MORTALE/QUAN
DO BELTA SPLENDEA/NEGLI OCCHI TU
OI RIDENTI E FUGGITIVI/E TU LIET
A E PENSOSA IL LIMITARE/DI GIOVE
NTU SALIVI

********************************

********************************

********************************
A:      SILVIA RIMEMBRI ANCORA/QUE
L TEMPO DELLA TUA VITA MORTALE/Q
UANDO BELTA SPLENDEA/NEGLI OCCHI
 TUOI RIDENTI E FUGGITIVI/E TU L
IETA E PENSOSA IL LIMITARE/DI GI
OVENTU SALIVI

********************************
E:      SILVIA RIMEMBRI ANCORA/QUE
L TEMPO DELLA TUA VITA MORTALE/Q
UANDO BELTA SPLENDEA/NEGLI OCCHI
 TUOI RIDENTI E FUGGITIVI/E TU L
IETA E PENSOSA IL LIMITARE/DI GI
OVENTU SALIVI

********************************
I:      SILVIA RIMEMBRI ANCORA/QUE
L TEMPO DELLA TUA VITA MORTALE/Q
UANDO BELTA SPLENDEA/NEGLI OCCHI
 TUOI RIDENTI E FUGGITIVI/E TU L
IETA E PENSOSA IL LIMITARE/DI GI
OVENTU SALIVI

********************************
O:      SILVIA RIMEMBRI ANCORA/QUE
L TEMPO DELLA TUA VITA MORTALE/Q
UANDO BELTA SPLENDEA/NEGLI OCCHI
 TUOI RIDENTI E FUGGITIVI/E TU L
IETA E PENSOSA IL LIMITARE/DI GI
OVENTU SALIVI

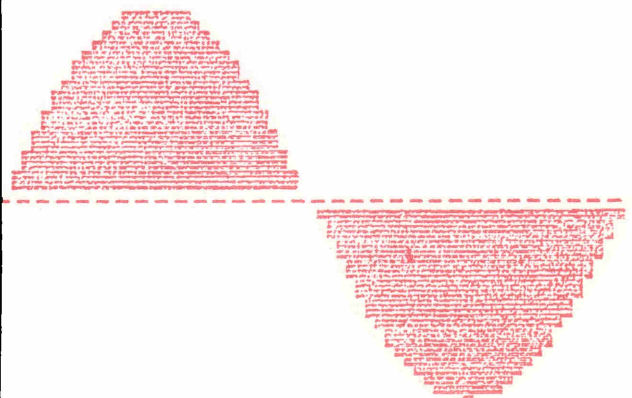
********************************
U:      SILVIA RIMEMBRI ANCORA/QUE
L TEMPO DELLA TUA VITA MORTALE/Q
UANDO BELTA SPLENDEA/NEGLI OCCHI
 TUOI RIDENTI E FUGGITIVI/E TU L
IETA E PENSOSA IL LIMITARE/DI GI
OVENTU SALIVI

********************************
```

```
8000 REM **FINE LAVORO**
8005 SLOW
8010 PRINT AT 21,0;"PREMI UN TASTO PER CONTINUARE..."
8020 IF INKEY$="" THEN GOTO 8020
8030 CLS
8035 GOSUB 30
8040 PRINT "SCEGLI:"
8050 PRINT
8060 PRINT "1 PER LAVORARE SU UN ALTRO TESTO"
8065 PRINT
8070 PRINT "2 PER UNA DIVERSA ANALISI","  DELLO STESSO TESTO"
8075 PRINT
8080 PRINT "3 PER FINIRE"
8090 GOSUB 30
8095 IF INKEY$<>"" THEN GOTO 8095
8100 IF INKEY$="" THEN GOTO 8100
8110 LET F=VAL INKEY$
8120 CLS
8130 FAST
8140 IF F=1 THEN RUN
8150 IF F=2 THEN GOTO 340
8160 PRINT AT 10,12;"[illegible]"
8990 STOP
9000 CLEAR
9010 SAVE "POE[illegible]"
9020 RUN
```

Seno

```
 1 REM ....................
 2 REM ......SINUS.........
 3 REM ....................
 4 REM ...TIM.HARTNELL.GB..
 5 REM ....................
10 FOR X=0 TO 63
20 LET Y=20*SIN (X/32*PI)
30 IF Y=0 THEN GOTO 70
40 FOR N=0 TO Y STEP SGN Y
50 PLOT X,N+22
60 NEXT N
70 NEXT X
80 PRINT AT 10,0;"--------------------------------"
```



Gli operatori logici

Tecniche di programmazione



AND e OR, nel BASIC del Sinclair, hanno due usi differenti. Il primo rispecchia il significato che le due parole hanno nel linguaggio comune ("e", "oppure"), ed è di facile comprensione.

Il secondo uso è meno lineare da comprendere, ma contiene una tecnica di programmazione molto versatile. NOT, il terzo operatore logico, è stranamente trascurato o ignorato da molti programmatori alle prime armi; tuttavia, è una funzione dalle molte possibilità.

AND E OR

```
IF A<10 AND B< 10 THEN ...
IF A<10  OR B  < 10 THEN ...
```

Nel primo esempio, l'istruzione che segue il THEN sarà eseguita se entrambe le condizioni sono *vere* (cioè se sia A che B sono minori di 10). Nel secondo caso, l'istruzione verrà eseguita se almeno una delle due istruzioni è vera.

AND ha priorità più alta di OR, ed è quindi affrontata per prima, indipendentemente dalla sua posizione nello statement condizionale. Per esempio, questa linea di programma

```
IF X> 5 OR Y >5 AND Z >5 THEN ...
```

viene suddivisa in:

a) X>5
b) Y>5 AND Z>5

poichè sono unite da un OR, basta che una delle due sia vera perchè l'istruzione che segue il THEN venga eseguita. Per modificare l'ordine di svolgimento delle operazioni che il computer segue automaticamente, basta usare delle parentesi:

```
IF (X> 5 OR Y> 5) AND Z > 5 THEN ...
```

Ora le due condizioni sono:

a) X > 5 OR Y> 5
b) Z > 5

Invece di IF/THEN

AND e OR possono essere usati in un altro modo, che permette di combinare più di un'istruzione IF/THEN in un'unica linea di programma, senza nemmeno che sia necessario usare lo statement IF/THEN. Per esempio:

```
IF A < 10 THEN GOTO 100
IF A = 10 THEN GOTO 200
IF A > 10 THEN GOTO 300
```

può essere riscritta così:

```
GOTO (100 AND A< 10)+(200 AND A = 10)
+ (300 AND A > 10)
```

Ogni espressione tra parentesi viene esaminata dal computer come un test *vero/falso*. I termini di sinistra, numerici, sono necessariamente *veri*, dei termini di destra è chiaramente *vero* solo uno dei tre. Alle espressioni contenenti i termini *falsi* viene assegnato valore logico zero. Per esempio, se A = 20, l'istruzione viene letta così:

```
GOTO (0) + (0) + (300)
```

Reciproca esclusione

Nell'esempio precedente, le tre condizioni di A si escludevano reciprocamente. Ciò non avviene sempre e si possono avere situazioni come quella che segue:

```
IF B < 10 THEN GOTO 100
IF B = 10 THEN GOTO 200
IF B < 20 THEN GOTO 300
```

Com'è evidente, B può essere contemporaneamente minore di 10 e di 20, o può essere uguale a 10 - e quindi ancora minore di 20. Istruzioni come queste possono essere disposte nel giusto ordine (giusto per il risultato che si vuole ottenere) e tutto andrà bene finchè verranno eseguite in quell'ordine.

Tuttavia, se provassimo a riscrivere quanto sopra usando gli operatori logici, ci troveremmo subito in difficoltà:

```
GOTO (100 AND B < 10) + (200 AND B=10) +
(300 AND B < 20)
```

dove, nel caso di B=10 si realizzano due condizioni contemporaneamente vere:

```
GOTO (0) + (200) + (300)
```

e il programma cercherà di eseguire GOTO 500!

E' necessaria, quindi, una modifica, in modo che tutte le condizioni fra parentesi si escludano reciprocamente:

```
IF B < 10 ...
IF B = 10...
IF B < 20 AND B  > 10 ...
```

che, con gli operatori logici, diventa:

```
GOTO (100 AND B < 10) + (200 AND B=10)+
(300 AND (B < 20 AND B > 10 ) )
```

Osserviamo che l'ultimo AND opera nel modo descritto nel primo paragrafo, a differenza di tutti gli altri. Le parentesi interne nell'ultima condizione non sono indispensabili: sono state inserite solo per maggiore chiarezza.

Altre istruzioni

L'operatore logico può essere utilizzato nel modo appena descritto in diverse situazioni:

```
PRINT AT (N AND X> 15)+ (M AND X< 10);A$
PLOT K, (7 AND X=13) + (37 AND X ≠ 13)
PAUSE (200 AND T=11) + (400 AND T>= 13)
etc.
```

Stringhe

Similmente, negli statement si possono trattare stringhe alfanumeriche: quando il risultato dell'operazione logica è *vero*, il contenuto delle parentesi diventa la stringa che precede l'AND. Se è *falso*, si ha una stringa nulla. Per esempio:

```
IF H> K THEN PRINT "TROPPO GRANDE"
IF H< K THEN PRINT "TROPPO PICCOLO"
```

può diventare

```
PRINT ("TROPPO GRANDE" AND H> K) +
("TROPPO PICCOLO" AND H < K)
```

che stamperà una delle due risposte, secondo il caso.

Un altro esempio:

```
LET A$ = ("GIUSTO" AND H = K) +
("SBAGLIATO"  AND H ≠ K)
```

L'uso dell'espressione logica non è limitato alla situazione di scelta tra due alternative: può trattarsi di una condizione singola. Così l'istruzione:

```
PRINT A$; TAB 12; ("*" AND K =< 15); TAB
18; N
```

stamperà un asterisco nella colonna 12 del video se è *vera* la condizione richiesta e niente in caso contrario.

Ridefinizione di variabili

E' facile anche usare le espressioni logiche per ridefinire il valore di una variabile. Ecco un esempio:

```
LET X= X + (10 AND H< K) – (7 AND H> K)
```

che eseguirà:

```
LET X =X + (10)-(0)     per H< K
LET X =X + (0) - (7)    per H> K
LET X =X + (0) - (0)    per H = K
```

Come si vede, se nessuna condizione è verificata, il valore della variabile non viene alterato.

OR

Mentre AND viene usato per operazioni di somma e sottrazione, OR è praticato per moltiplicazioni o divisioni. Studiatevi questa tabella:

operatore	scelta condiz.	espress.	risultato
AND	A$	*vero*	A$
AND	A	*falso*	" " (stringa nulla)
AND	N	*vero*	N
AND	N	*falso*	O
OR	N	*vero*	1
OR	N	*falso*	N

Ecco uno dei modi di usare OR (che qui potremmo leggere come "a meno che..."):

```
LET X = X * (10 OR A > B)
```

Se A>B è *vero*, all'espressione fra parentesi viene assegnato valore 1 e il valore di X non cambia:

```
LET X = X * (1)
```

Se A > B è *falso*, l'espressione fra parentesi assume il valore che precede l'OR:

```
LET X = X * (10)
```

Quindi, X viene moltiplicato per 10, "a meno che" A sia maggiore di B, caso in cui resta invariato (notate che se avessimo usato AND ci saremmo trovati, nel caso di condizione *falsa*, con la variabile azzerata). Un altro esempio:

```
LET P = P * (1.35 OR C3 < 2500) * (1.18 OR
C3 > = 2500)
```

che fornisce il prezzo "ivato" di un'automobile, con la condizione che l'IVA passa dal 18% al 35% quando la cilindrata supera i 2500 cmc. Il test funziona così:

```
se C3 < 2500 ,   P = P * (1) * (1.18)
se C3 > = 2500,  P = P * (1.35) * (1)
```

Istruzioni non esclusive

A volte può esserci l'esigenza che diverse espressioni logiche, nella ridefinizione di una variabile, non siano reciprocamente esclusive. Consideriamo un gioco in cui si ricevono 100 punti ogni ciclo "positivo" (i soliti UFO da abbattere, per esempio), diciamo ogni 5 colpi andati a segno, e un 'bonus' di 200 punti se i 5 colpi sono consecutivi (senza errori intermedi): si potrebbe avere un'istruzione con il seguente formato:

```
LET SCORE = SCORE + (100 AND H > 4) +
(200 AND K > 4)
```

dove H e K contano entrambi i colpi riusciti (e

vengono azzerati ogni 5), con la differenza che K viene rimesso a zero anche ad ogni colpo a vuoto.

Lo stesso discorso può essere fatto per gli OR. Consideriamo un altro esempio, che potrebbe essere ancora il punteggio di un gioco:

LET SC = SC * (10 OR H < 5) * (10 OR H< 10) * (10 OR H < 15)

Quando H< 5, si ha il *vero* su tutta la linea, e SC non cambia:

LET SC = SC * (1) * (1) * (1)

Con H tra 5 e 10, tra 10 e 15, maggiore di 15, il punteggio è moltiplicato rispettivamente per 10, 100, 1000.

NOT

Per usare correttamente NOT, bisogna comprendere quello che è, per il computer, il concetto di *vero/falso*. Quando analizza le condizioni di un'istruzione IF/THEN, assegna internamente "1" a una condizione *vera*, "0" ad una *falsa*.

Inoltre, qualsiasi espressione matematica, il cui risultato sia zero, è considerata logicamente *falsa*, mentre un risultato diverso da zero, anche negativo, è *vero*. NOT cambia il valore *vero/falso* di un'espressione:

se A è *falso*, NOT A è *vero;*
se A è *vero*, NOT A è *falso.*

Uso del NOT

Poniamo di avere in un loop un'istruzione da eseguire solo negli ingressi pari (una volta sì e una no, a partire dal secondo ingresso). Controlleremo l'esecuzione con una linea di questo tipo:

IF K THEN . . (comando)

dove K è una variabile inizializzata, diversa da zero. All'uscita dal loop, inseriamo:

LET K = NOT K

che "inverte" ogni volta il valore logico di K.

Altro caso:

LET A = N/2 - INT (N/2)

A è zero se N è un numero pari. Con l'istruzione

IF NOT A THEN . . .

possiamo controllare operazioni da eseguire solo quando N è pari. Allo stesso modo,

LET I = N–INT N

assegna I=O quando N è un intero, e facendo seguire

IF NOT I THEN . . .

si abilita l'esecuzione di quanto segue il THEN, solo se N è un numero intero.

Priorità

NOT opera soltanto sul primo numero alla sua destra a differenza degli altri operatori logici, che intervengono sull'intera espressione. Così:

NOT B < C

scritto senza parentesi, viene interpretato come:

(NOT B) < C

NOT ha inoltre la priorità più alta sia di AND che di OR: l'espressione

IF NOT A AND B OR C

viene letta dal computer come se fosse

IF ((NOT A) AND B) OR C

Naturalmente, tale priorità può essere modificata usando le parentesi. ■

• **Per controllare lo SCROLL** in un programma, in cui state scrivendo sul video dei dati qualsiasi, il sistema più efficace è il seguente:

IF PEEK 16442 <= 3 THEN SCROLL

In questa locazione di memoria si trova la variabile di sistema che controlla il codice di errore 5 (schermo pieno).

Provate il funzionamento, per esempio così:

```
10 FOR L=0 TO 63
20 PRINT CHR$ L
30 IF PEEK 16442<=3 THEN SCROL
L
40 NEXT L
```

ESCLUSIVO

LA ROM dello ZX81

(prima parte)

```
0000 D3 FD          OUT (FD),A
0002 01 FF 7F       LD BC,7FFF
0005 C3 CB 03       JP 03CB
```

ERRORE/1

Tutti i messaggi di errore sono sotto il controllo di queste istruzioni, associate alla chiamata del sottoprogramma ERRORE/2.

in HL la variabile di sistema CH-ADD (v.)

salta alla rout. ERRORE/2

```
0008 2A 16 40       LD HL,(4016)
000B 22 18 40       LD (4018),HL
000E 18 46          JR 0056
```

PRINT/1

Controlla la rappresentazione dei caratteri. Il carattere da trattare si trova nel registro A.

salta al simbolo PRINT
salta alla funzione PRINT
valore 255

```
0010 A7             AND A
0011 C2 F1 07       JP NZ,07F1
0014 C3 F5 07       JP 07F5
0017 FF             DB
```

Va a prendere un carattere

Carica il reg. A col valore della locazione indirizzata da CH-ADD. I caratteri vuoti vengono ignorati.

CH-ADD in HL
valore di CH-ADD nell'accumulatore

```
0018 2A 16 40       LD HL,(4016)
001B 7E             LD A,(HL)
001C A7             AND A
001D C0             RET NZ
001E 00             NOP
001F 00             NOP
```

Va a prendere il carattere successivo

CH-ADD viene incrementato, prima di andare a mettere un carattere nel reg. A. Anche qui i vuoti vengono ignorati.

```
0020 CD 49 00      CALL 0049     chiama la sbr. 0049
0023 18 F7         JR 001C       salta a 001C
0025 FF            DB            valore 255
0026 FF            DB
0027 FF            DB
```

ARITMETICA

Chiamata della sezione aritmetica della ROM.

```
0028 C3 9D 19      JP 199D       salta a 199D
```

FINE ARITMETICA

Lascia la sez. Aritmetica e nel puntatore della catasta (reg. SP - stack pointer) viene ripristinato il valore precedente.

```
002B F1            POP AF
002C D9            EXX
002D E3            EX (SP),HL
002E D9            EXX
002F C9            RET
```

SPAZIO

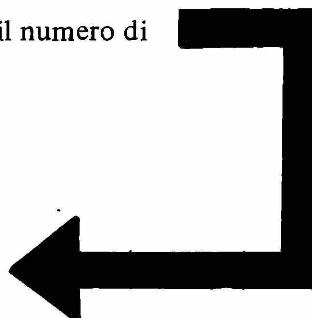
Produce posto di memoria libero nell'area di lavoro. Il numero di bytes liberi si trova nei registri BC.

```
0030 C5            PUSH BC           BC nello stack
0031 2A 14 40      LD HL,(4014)      carica E-line in HL (°)
0034 E5            PUSH HL
0035 C3 88 14      JP 1488           salta a 1488
```

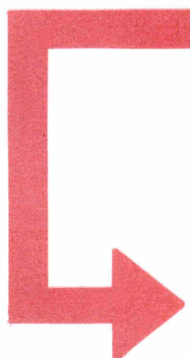
(°) E-line segna la fine dell'area variabili; v. manuale.

INT

Interrupt che serve per la gestione dei video. Nel reg. B c'è il numero di linea e in C la quantità delle linee.



```
0038 0D          DEC C
0039 C2 45 00    JP NZ,0045
003C E1          POP HL
003D 05          DEC B
003E C8          RET Z
003F CB D9       SET 3,C
0041 ED 4F       LD R,A
0043 FB          EI
0044 E9          JP (HL)
0045 D1          POP DE
0046 C8          RET Z
0047 18 F8       JR 0041
```



CH-ADD + 1

Incrementa di 1 il puntatore CH-ADD. Nel caso che questo puntatore arrivi sopra il cursore, rappresentato da 7F, viene incrementato di nuovo.

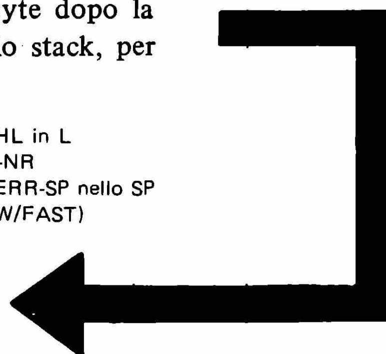
CH-ADD in HL
incr.
HL in CH-ADD
valore di HL nell'accum.
confronto col cursore

```
0049 2A 16 40    LD HL,(4016)
004C 23          INC HL
004D 22 16 40    LD (4016),HL
0050 7E          LD A,(HL)
0051 FE 7F       CP 7F
0053 C0          RET NZ
0054 18 F6       JR 004C
```

ERRORE/2

Determina il codice di errore, che si trova sempre un byte dopo la chiamata. Quindi diventa l'indirizzo di return, preso dallo stack, per conservarlo in questo byte.

valore di HL in L
L in ERR-NR
valore di ERR-SP nello SP
(test SLOW/FAST)



```
0056 E1          POP HL
0057 6E          LD L,(HL)
0058 FD 75 00    LD (IY+00),L
005B ED 7B 02 40 LD SP,(4002)
005F CD 07 02    CALL 0207
0062 C3 BC 14    JP 14BC
0065 FF          DB
```

COMPOSIZIONE DELL'IMMAGINE

Esegue un test per verificare se il computer si trovi in modo FAST o SLOW, interrogando il bit 6 del flag CD. Se è attivato il modo FAST, non viene formata nessuna immagine e si esce dalla routine. Diversamente, il programma prosegue con la prossima sezione.

```
0207 21 3B 40     LD HL,403B
020A 7E           LD A,(HL)
020B 17           RLA
020C AE           XOR(HL)
020D 17           RLA
020E D0           RET NC
020F 3E 7F        LD A,7F
0211 08           EX AF,AF
0212 06 11        LD B,11
0214 D3 FE        OUT(FE),A
0216 10 FE        DJNZ 0216
0218 D3 FD        OUT(FD),A
021A 08           EX AF,AF
021B 17           RLA
021C 30 08        JR NC,0226
021E CB FE        SET 7,(HL)
0220 F5           PUSH AF
0221 C5           PUSH BC
0222 D5           PUSH DE
0223 E5           PUSH HL
0224 18 03        JR 0229
0226 CB B6        RES 6,(HL)
0228 C9           RET
```

COMPOSIZIONE DELL'IMMAGINE/1

Anzitutto viene decrementato il contatore di fotogrammi (FRAMES, 16436). Se è zero, si ritorna al programma principale. Questa routine è attivata, per esempio, da un'istruzione di pausa.

```
0229 2A 34 40     LD HL,(4034)
022C 2B           DEC HL
022D 3E 7F        LD A,7F
022F A4           AND H
0230 B5           OR L
0231 7C           LD A,H
0232 20 03        JR NZ,0237
0234 17           RLA
0235 18 02        JR 0239
0237 46           LD B,(HL)
0238 37           SCF
0239 67           LD H,A
023A 22 34 40     LD(4034),HL
023D D0           RET NC
```

COMPOSIZIONE DELL'IMMAGINE/2

Legge la tastiera e, nel caso che sia stato premuto un tasto, esce da questa sezione di programma; altrimenti si ha un'immagine sul video.

- interroga la tastiera.
- valore di LAST-K (ultimo tasto premuto)
- HL in LAST-K
- valore di (16423) nell'accumulatore. (=stato tastiera)
- salta, se nessun tasto è stato premuto
- stato tastiera in HL
- contenuto di D-FILE in HL

```
023E CD BB 02       CALL 02BB
0241 ED 4B 25 40    LD BC,(4025)
0245 22 25 40       LD (4025),HL
0248 78             LD A,B
0249 C6 02          ADD A,02
024B ED 42          SBC HL,BC
024D 3A 27 40       LD A,(4027)
0250 B4             OR H
0251 B5             OR L
0252 58             LD E,B
0253 06 0B          LD B,0B
0255 21 3B 40       LD HL,403B
0258 CB 86          RES 0,(HL)
025A 20 08          JR NZ,0264
025C CB 7E          BIT 7,(HL)
025E CB C6          SET 0,(HL)
0260 C8             RET Z
0261 05             DEC B
0262 00             NOP
0263 37             SCF
0264 21 27 40       LD HL,4027
0267 3F             CCF
0268 CB 10          RL B
026A 10 FE          DJNZ 026A
026C 46             LD B,(HL)
026D 7B             LD A,E
026E FE FE          CP,FE
0270 9F             SBC A,A
0271 06 1F          LD B,1F
0273 B6             OR (HL)
0274 A0             AND B
0275 1F             RRA
0276 77             LD (HL),A
0277 D3 FF          OUT (FF),A
0279 2A 0C 40       LD HL,(400C)
027C CB FC          SET 7,H
027E CD 92 02       CALL 0292
0281 ED 5F          LD A,R
0283 01 01 19       LD BC,1901
0286 3E F5          LD A,F5
0288 CD B5 02       CALL 02B5
028B 2B             DEC HL
028C CD 92 02       CALL 0292
028F C3 29 02       JP 0229
```

COMPOSIZIONE DELL'IMMAGINE/3

Alla fine della composizione dell'immagine, questo sottoprogramma riporta l'indirizzo di ritorno in IX. Le coppie AF, BC, DE, HL vengono restituite con il valore iniziale.

```
0292 DD E1          POP IX
0294 FD 4E 28       LD C,(IY+28)
0297 FD CB 3B 7E    BIT 7,(HL)
029B 28 0C          JR Z,02A9
029D 79             LD A,C
029E ED 44          NEG
02A0 3C             INC A
02A1 08             EX AF,AF
02A2 D3 FE          OUT (FE),A
02A4 E1             POP HL
02A5 D1             POP DE
02A6 C1             POP BC
02A7 F1             POP AF
02A8 C9             RET
```

COMPOSIZIONE DELL'IMMAGINE/4

I registri A e B vengono caricati con determinati valori, per rendere possibile la formazione dell'immagine sul video.

```
02A9  3E FC        LD A,FC
02AB  06 01        LD B,01
02AD  CD B5 02     CALL 02B5
02B0  2B           DEC HL
02B1  E3           EX (SP),HL
02B2  E3           EX (SP),HL
02B3  DD E9        JP (IX)
```

COMPOSIZIONE DELL'IMMAGINE/5

Il registro R (refresh della memoria) viene caricato, mentre si esegue un interrupt.

```
02B5  ED 4F        LD R,A
02B7  3E DD        LD A,DD
02B9  FB           EI
02BA  E9           JP (HL)
```

INTERVENTI DI TASTIERA

Sottoprogramma molto importante e di frequente uso. La tastiera viene interrogata otto volte con differenti indirizzi di chiamata, per verificare quale tasto sia stato premuto. Ad ogni tasto corrisponde un valore nella coppia di registri HL. Se nessun tasto è stato premuto, in HL entra il valore FFFF. Inoltre si determina il valore di MARGIN (v.).

```
02BB  21 FF FF     LD HL,FFFF     Interroga la tastiera
02BE  01 FE FE     LD BC,FEFE     riconosce il primo indi-
02C1  ED 78        IN A,(C)       rizzo di tastiera
02C3  F6 01        OR 01
02C5  F6 E0        OR E0          bit 0 settato
02C7  57           LD D,A         bit 5/6/7 settati
02C8  2F           CPL            °il valore di tast. letto
02C9  FE 01        CP 01          è trasformato in modo che
02CB  9F           SBC A,A        HL possa memorizzare gli 8
02CC  B0           OR B           diversi valori
02CD  A5           AND L
02CE  6F           LD L,A
02CF  7C           LD A,H         Se nessun tasto è premuto,
02D0  A2           AND D          HL non è modificato.
02D1  67           LD H,A
02D2  CB 00        RLC B
02D4  ED 78        IN A,(C)       prossimo ind. di tast.
02D6  38 ED        JR C,02C5
02D8  1F           RRA
02D9  CB 14        RL H           test per le linee di schermo;
02DB  17           RLA            è diverso se il video
02DC  17           RLA            collegato è a norma europea
02DD  17           RLA            o americana. Il controllo
02DE  9F           SBC A,A        avviene sul piedino 22 di IC1.
02DF  E6 18        AND 18
02E1  C6 1F        ADD A,1F
02E3  32 28 40     LD (4028),A
02E6  C9           RET
```

NMI

Viene chiamata quando la CPU contiene un segnale NMI. Ciò accade quando viene attivato il modo SLOW.

```
0066 08          EX AF,AF
0067 3C          INC A
0068 FA 6D 00    JP M,006D
006B 28 02       JR Z,006F
006D 08          EX AF,AF
006E C9          RET
```

NMI/1

Viene chiamata da NMI (v.), e prepara un'uscita sullo schermo TV. Le coppie di regg. AF, BC, DE, HL vengono spinte nello stack, il generatore NMI è disattivato e si chiama la subr. di uscita sul video (il reg. IX in questo momento ha il valore 0281 oppure 028F).

```
006F 08          EX AF,AF
0070 F5          PUSH AF
0071 C5          PUSH BC
0072 D5          PUSH DE
0073 E5          PUSH HL
0074 2A 0C 40    LD HL,(400C)
0077 CB FC       SET 7,H
0079 76          HALT
007A D3 FD       OUT (FD),A
007C DD E9       JP (IX)
```

La tabella dei tasti

Da 007E a 00A4	caratteri senza SHIFT
da 00A5 a 00CB	caratteri con SHIFT
da 00CC a 00F2	caratteri in modo F
da 00F3 a 0110	caratteri in modo G
da 0111 a 01FB	tabella delle parole "compresse" per le istruzioni. Per ogni istruzione si riporta il codice memorizzato; il sistema mantiene i codici dei singoli caratteri con cui è scritta l'istruzione. L'ultimo carattere di ciascuna parola è 'inverse' per riconoscerne la fine.

```
007E 3F 63    Z        0099 3A 58    U
007F 3D 61    X        009A 3E 62    Y
0080 28 40    C        009B 76 118   ?
0081 3B 59    V        009C 31 49    L
0082 26 38    A        009D 30 48    K
```

0083	38	56	S
0084	29	41	D
0085	2B	43	F
0086	2C	44	G
0087	36	54	Q
0088	3C	60	W
0089	2A	42	E
008A	37	55	R
008B	39	57	T
008C	1D	29	1
008D	1E	30	2
008E	1F	31	3
008F	20	32	4
0090	21	33	5
0091	1C	28	0
0092	25	37	9
0093	24	36	8
0094	23	35	7
0095	22	34	6
0096	35	53	P
0097	34	52	O
0098	2E	46	I

009E	2F	47	J
009F	2D	45	H
00A0	00	0	
00A1	1B	27	.
00A2	32	50	M
00A3	33	51	N
00A4	27	39	B
00A5	0E	14	:
00A6	19	25	;
00A7	0F	15	?
00A8	18	24	/
00A9	E3	227	STOP
00AA	E1	225	LPRINT
00AB	E4	228	SLOW
00AC	E5	229	FAST
00AD	E2	226	LLIST
00AE	C0	192	""
00AF	D9	217	OR
00B0	E0	224	STEP
00B1	DB	219	<=
00B2	DD	221	<>
00B3	75	117	?
00B4	DA	218	AND

00B5	DE	222	THEN
00B6	DF	223	TO
00B7	72	114	?
00B8	77	119	?
00B9	74	116	?
00BA	73	115	?
00BB	70	112	?
00BC	71	113	?
00BD	0B	11	"
00BE	11	17	)
00BF	10	16	(
00C0	0D	13	$
00C1	DC	220	>=
00C2	79	121	?
00C3	14	20	=
00C4	15	21	+
00C5	16	22	-
00C6	D8	216	**
00C7	0C	12	£
00C8	1A	26	,
00C9	12	18	>
00CA	13	19	<
00CB	17	23	*
00CC	CD	205	LN
00CD	CE	206	EXP
00CE	C1	193	AT
00CF	78	120	?
00D0	CA	202	ASN
00D1	CB	203	ACS
00D2	CC	204	ATN
00D3	D1	209	SGN
00D4	D2	210	ABS
00D5	C7	199	SIN
00D6	C8	200	COS
00D7	C9	201	TAN
00D8	CF	207	INT
00D9	40	64	RND
00DA	78	120	?
00DB	78	120	?
00DC	78	120	?
00DD	78	120	?
00DE	78	120	?
00DF	78	120	?
00E0	78	120	?
00E1	78	120	?
00E2	78	120	?

00E3	78	120	?
00E4	C2	194	TAB
00E5	D3	211	PEEK
00E6	C4	196	CODE
00E7	D6	214	CHR$
00E8	D5	213	STR$
00E9	78	120	?
00EA	D4	212	USR
00EB	C6	198	LEN
00EC	C5	197	VAL
00ED	D0	208	SQR
00EE	78	120	?
00EF	78	120	?
00F0	42	66	PI
00F1	D7	215	NOT
00F2	41	65	INKEY$
00F3	08	8	
00F4	0A	10	
00F5	09	9	
00F6	8A	138	
00F7	89	137	
00F8	81	129	
00F9	82	130	
00FA	07	7	
00FB	84	132	
00FC	06	6	
00FD	01	1	
00FE	02	2	
00FF	87	135	
0100	04	4	
0101	05	5	
0102	77	119	?
0103	78	120	?
0104	85	133	
0105	03	3	
0106	83	131	
0107	8B	139	
0108	91	145	
0109	90	144	
010A	8D	141	
010B	86	134	
010C	78	120	?
010D	92	146	
010E	95	149	
010F	96	150	
0110	88	136	

```
0170 31 35 37 2E 33 B9   LPRINT      0111 8F                ?
0176 31 31 2E 38 B9   LLIST          0112 0B 8B             ""
017B 38 39 34 B5   STOP              0114 26 B9             AT
017F 38 31 34 BC   SLOW              0116 39 26 A7          TAB
0183 2B 26 38 B9   FAST              0119 8F                ?
0187 33 2A BC      NEW               011A 28 34 29 AA       CODE
018A 38 28 37 34 31 B1   SCROLL      011E 3B 26 B1          VAL
0190 28 34 33 B9   CONT              0121 31 2A B3          LEN
0194 29 2E B2      DIM               0124 38 2E B3          SIN
0197 37 2A B2      REM               0127 28 34 B8          COS
019A 2B 34 B7      FOR               012A 39 26 B3          TAN
019D 2C 34 39 B4   GOTO              012D 26 38 B3          ASN
01A1 2C 34 38 3A A7   GOSUB          0130 26 28 B8          ACS
01A6 2E 33 35 3A B9   INPUT          0133 26 39 B3          ATN
01AB 31 34 26 A9   LOAD              0136 31 B3             LN
01AF 31 2E 38 B9   LIST              0138 2A 3D B5          EXP
01B3 31 2A B9      LET               013B 2E 33 B9          INT
01B6 35 26 3A 38 AA   PAUSE          013E 38 36 B7          SQR
01BB 33 2A 3D B9   NEXT              0141 38 2C B3          SGN
01BF 35 34 30 AA   POKE              0144 26 27 B8          ABS
01C3 35 37 2E 33 B9   PRINT          0147 35 2A 2A B0       PEEK
01C8 35 31 34 B9   PLOT              014B 3A 38 B7          USR
01CC 37 3A B3      RUN               014E 38 39 37 8D       STR$
01CF 38 26 3B AA   SAVE              0152 28 2D 37 8D       CHR$
01D3 37 26 33 A9   RAND              0156 33 34 B9          NOT
01D7 2E AB         IF                0159 17 97             **
01D9 28 31 B8      CLS               015B 34 B7             OR
01DC 3A 33 35 31 34 B9   UNPLOT      015D 26 33 A9          AND
01E2 28 31 2A 26 B7      CLEAR       0160 13 94             <=
01E7 37 2A 39 3A 37 B3   RETURN      0162 12 94             >=
01ED 28 34 35 BE         COPY        0164 13 92             <>
01F1 37 33 A9            RND         0166 39 2D 2A B3       THEN
01F4 2E 33 30 2A 3E 8D   INKEY$      016A 39 B4             TO
01FA 35 AE               PI          016C 38 39 2A B5       STEP
```

UPDATE

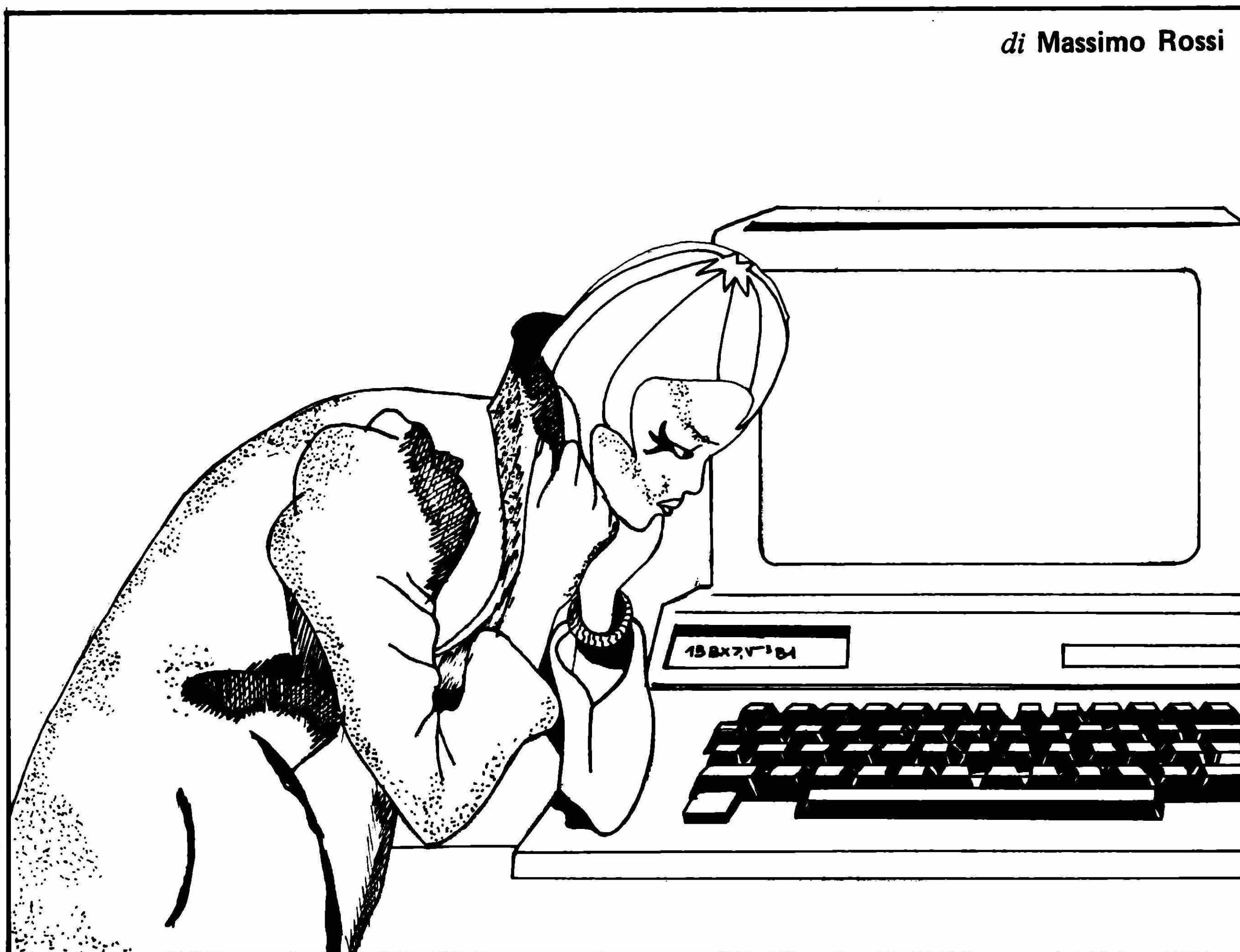
Sottoprogramma usato con LOAD e SAVE. Il registro HL viene incrementato finchè arriva a E-line. HL presenta, a seconda del caso, il byte che deve essere caricato o memorizzato.
Nota: la routine prepara il byte successivo. Nella coppia di regg. HL entra il valore di E-line per verificare se il programma è già alla fine. Con LOAD viene caricato E-line, con le variabili di sistema, all'inizio del programma. Il valore impegnato all'inizio di un LOAD è pari a una lunghezza di programma di 256 bytes. Unitamente viene trasmesso l'indirizzo di ritorno.

```
01FC 23           INC HL
01FD EB           EX DE,HL
01FE 2A 14 40     LD HL,(4014)
0201 37           SCF
0202 ED 52        SBC HL,DE
0204 EB           EX DE,HL
0205 D0           RET NC
0206 E1           POP HL
```

Chi c'é dietro il beep?

Intercettiamo i discorsi tra lo Spectrum e il nostro registratore.

di **Massimo Rossi**



CHIUNQUE abbia caricato su nastro dei dati sotto forma di byte, si sarà accorto che, una volta registrati, quando si desidera ricaricarli, non è più possibile sapere da dove iniziava la zona di memoria che si era "salvata", nè è possibile sapere di quanti byte si trattava. Questo problema è ancora più sentito da chi, essendo in possesso di qualche gioco, o altro programma, scritto in codice macchina, può caricare tranquillamente tutti i byte, ma non può farne alcuna copia, dato che lo Spectrum richiede un comando di SAVE, corredato dall'istruzione CODE, seguita dall'indirizzo di start, e dal numero di byte.

Come possiamo, dunque, procurarci queste informazioni?

Ovviamente il computer, quando carica un programma da nastro, deve venir informato su quanti byte stanno per arrivargli e dove dovrà metterli. Queste informazioni, gli pervengono dalla lettura di un messaggio speciale, chiamato "header".

Infatti, se ascoltiamo un nastro registrato con un programma, sentiremo, prima un suono lungo tutto uguale, (questo è un segnale di sincronismo, che permette al computer di sintonizzarsi col nastro), quindi, una scarica di suoni aspri, molto breve. Questo è l'header. Sentiremo, successivamente, una pausa seguita da una lunga scarica di suoni: si tratta del programma vero e proprio, sotto forma di byte.

Perciò, se riuscissimo a leggere il messaggio iniziale, sapremmo di che tipo di programma si tratta, da dove incomincia, e di quanti byte è composto.

La possibilità di leggere l'header, esiste, ma è necessario utilizzare le routine del sistema operativo, per cui servirà un po' (ma solo un po'), di linguaggio macchina.

Prima di passare al programma, diamo un'occhiata a questo header, al fine di imparare qualcosa di nuovo e di utile sulla nostra macchina.

L'header è composto da 17 byte in tutto, suddivisi nel seguente modo (vedi tabella 1).

- Il primo byte segnala il tipo di dati che seguirà:

00 Programma basic
01 Array numerico
02 Array alfanumerico
03 Blocco di byte

- I successivi 10 byte contengono il nome del programma, scritto in codice ASCII; se il nome del programma è più breve di 10 caratteri, i restanti byte, vengono riempiti con degli "space".
- Due byte contengono il numero dei byte che stanno per essere mandati.
- Altri due byte, che per un programma basic, contengono la linea di autostart (LINE), mentre per un blocco di bytes sono l'indirizzo di start.
- Questi ultimi byte esistono solo nel caso di un programma basic, e contengono la lunghezza del programma senza le variabili eventuali.

La struttura del nostro programma consisterà nel programmino in linguaggio macchina, che ci fa passare attraverso la routine che lo stesso Spectrum usa per leggere gli header, ma con una piccola modifica: gli faremo porre il risultato della lettura in un luogo diverso da quello uguale,

BASIC	Numero bytes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	Lettura esadecimale	00	41	54	54	41	43	4B	20	20	20	20	2A	0F	0A	00	1A	0F
	Lettura decimale	00	65	84	84	65	67	75	32	32	32	32	42	15	10	00	26	15
	Significato	Basic	A	T	T	A	C	K	Space	Space	Space	Space	Numero Bytes ← Variabili 3882		Linea Start . Line 10		Numero bytes senza variabili 3866	
BYTES	Numero bytes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	Lettura esadecimale	03	53	54	41	52	54	52	45	4B	20	20	A8	61	C0	5D	//	//
	Lettura decimale	03	83	84	65	82	84	82	69	75	32	32	168	97	192	93	//	//
	Significato	Byte	S	T	A	R	T	R	E	K	Space	Space	Numero bytes 25.000		Start address 24.000		Non utilizzati	

Si confrontano gli headers di due tipi di lettura da nastro: il primo è un programma in BASIC, con inizio automatico alla riga 10 (sopra); il secondo è un numero di BYTES (25.000), che dovranno essere caricati a partire dall'indirizzo decimale 24.000 (esadecimale 5DC0).

```
  10 REM ***************
  20 FOR i=23760 TO 23772: READ
 a: POKE i,a: NEXT i
  30 DATA 221,33,4,91,17,17,0,17
5,55,205,86,5,201
  40 CLS : PRINT "Inserire la ca
ssetta da analizzare nel registr
atore, e premere PLAY"
  50 PRINT USR 23760
  60 IF PEEK (23300)>0 THEN GO T
O 120
  70 CLS : PRINT "Programma in B
asic"
  80 GO SUB 1000: PRINT : PRINT
"Titolo: ";a$
  90 LET a=PEEK (23313)+PEEK (23
314)*256: IF a>9999 THEN GO TO 1
00
  95 PRINT : PRINT "LINE: ";a
 100 PRINT : PRINT "Lunghezza Pr
ogramma: ";PEEK (23311)+PEEK (23
312)*256: STOP
 120 CLS : PRINT "Bytes"
 130 GO SUB 1000: PRINT : PRINT
"Titolo: ";a$
 140 PRINT : PRINT "START: ";PEE
K (23313)+PEEK (23314)*256
 150 PRINT : PRINT "Numero Bytes
: ";PEEK (23311)+PEEK (23312)*25
6: STOP
1000 LET a$="": FOR i=23301 TO 2
3310: LET a$=a$+CHR$ PEEK (i): N
EXT i: RETURN
```

che in genere è il "workspace", spazio libero subito dopo l'area occupata dal basic.

Abbiamo scelto un'area della memoria che non dobbiamo utilizzare nel nostro programma: si tratta dell'area "Printer Buffer", utilizzata dalla stampante, ma non nel nostro programma; essa inizia all'indirizzo 23296, ma noi la utilizzeremo da 23300.

Una volta scritto l'header, il nostro programma si preoccuperà di definire di che tipo di dati si tratta, leggendo la prima locazione (23300), e quindi salterà alla opportuna routine che interpreterà i successivi dati, cioè la lunghezza e l'inizio dei dati in memoria, per un blocco di byte, e l'eventuale linea di autostart per un programma basic.

Due parole sulla routinetta in linguaggio macchina.

Essa viene caricata automaticamente dal programma ogni volta che gli si dà il RUN, nello spazio da noi creato in linea 10. Qui, infatti, dovremo digitare, dopo il REM, un numero di asterischi, o di qualsiasi altro carattere, sufficiente a creare spazio per la nostra routine.

Poichè si tratta di un programma che occupa 15 byte, batteremo 15 asterischi, assicurandoci così, uno spazio libero in memoria che l'interprete basic non leggerà.

Dobbiamo ricordare che, una volta dato il RUN, non potremo più listare il programma se non a partire dalla riga 20, e che ogni riga da correggere andrà richiamata col proprio numero, prima di essere editata. Perciò si suggerisce di registrare il programma prima di dare il RUN, anche per evitare eventuali errori di digitazione, mandiño il computer in reset, obbligandoci a ribattere tutto.

Per chi volesse approfondire il discorso del linguaggio macchina, riportiamo la routine per intero:

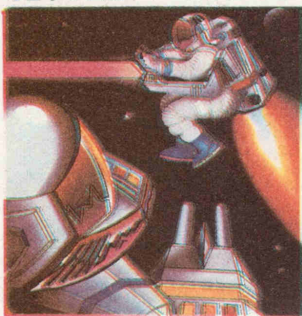
```
LD DE, 17
LD IX,23300
XOR A
SCF
CALL 1366
RET
```

In poche parole il programma richiama la routine del sistema operativo a 1366 decimale, cioè la routine "LOAD". E' essenziale, prima di richiamarla, che si sia caricato il registro DE, con il numero di byte che vogliamo caricare (nel nostro caso, 17), e il registro IX, con l'indirizzo da cui inizieremo a caricare (cioè 23300). Poi andrà caricato l'accumulatore, con 100 (XOR A), e settato il Carry Flag (altrimenti si esegue un Verify).

Una volta eseguita questa "routinetta", avremo l'header caricato da 23300, e potremo analizzarlo col programma in basic.

Quindi, concludendo, dovremo solo dare il RUN, far partire il registratore, e vedremo apparire le righe di LOAD; a questo punto, avremo sullo schermo, tutte le informazioni che desideravamo: il tipo di dati, la lunghezza, e la riga di autostart, se c'è. ▪

JET PAC

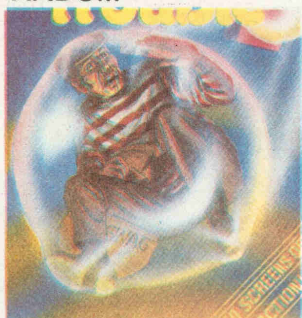


L'''Acme Interstellare'' sta trasportando un carico di componenti per astronavi verso vari pianeti del sistema solare. Come capo dei piloti collaudatori il giocatore deve assiemare i missili e sperare di raggiungere presto la prossima tappa.

Ma non sempre riuscite a scroccare un viaggio gratis, perciò per pagare il salatissimo biglietto delle astronavi, dovete raccogliere le pietre preziose che abbondano nei vari pianeti. Ciò vi dà l'occasione di arricchire oltre ogni misura.

Sistema richiesto: *Spectrum da 16 o 48K*
Editore: *Ultimate*
Distribuzione: *Rebit*

XADOM



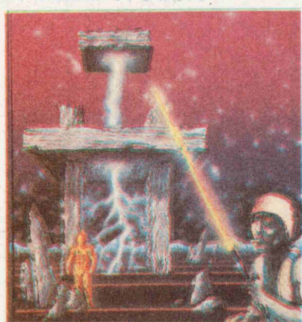
Tutto comincia un tardo pomeriggio d'estate. Voi siete sdraiati sul sofà, immersi nella visione dell'ultimo teleromanzo trasmesso dalla locale teleSensor-Visione ed aspettate l'arrivo della bellissima Luna Volupta. Ma al suo posto si fa avanti il vostro comandante, Grant Houseproud.

In breve, dovete abbandonare il vostro film, staccare dal ricaricatore la tuta biotronica e dirigervi in tutta fretta verso la sala delle riunioni del locale gran consiglio.

L'avventura comincia qui.

Sistema richiesto: *Spectrum*
Editore: *Quick Silva*
Distribuzione: *Rebit.*

Bubble Trouble

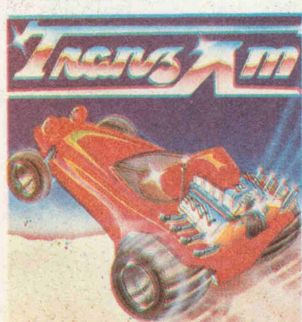


E' un videogioco tipo bar con una risoluzione elevata ed una strategia originale. Buono l'uso del suono e del colore.

Il giocatore è un maniaco che deve rubare tutto ciò che incontra nel corso delle 50 schermate di cui si compone la partita, ma una serie di mostri a forma dibolla o palloncino (da qui il titolo bubble troable, guaio della bolla) ti inseguono e cercano di soffocarti. Per liberarti puoi sparare un colpo con il joystick o battendo il tasto zero, ma questo bottone è attivo solo ogni 10 secondi.

Sistema richiesto: *ZX Spectrum da 48K*
Editore: *Arcade*
Distribuzione: *Rebit*

TRANZ AM



E' l'anno 3472 e quello che fu un pianeta vivo e lussureggiante, ora è una distesa desolata di rocce e di sabbia. Povera Terra! Ma la vita si adatta rapidamente, sebbene creando valori diversi: ad esempio, per essere qualcuno bisogna saper ritrovare le otto ''Grandi coppe'', rimaste sepolte per secoli dopo essere state possedute dai ''Deadly Black Turbos''.

Il gioco consiste nell'evitare di cozzare contro i Turbo, e nel riuscire a localizzare le otto Grandi coppe.

Sistema: *Spectrum*
Editore: *Ultimate play the game*
Distribuzione: *Rebit*

Se vuoi abbonarti

Registrate il mio abbonamento annuale a Sinclair Computer.

☐ Ho versato oggi stesso il canone di Lire 20.000 a mezzo c/c postale nº 30426209 intestato a: Sinclair Computer - V.le Famagosta, 75 - 20142 Milano

☐ Accludo assegno per lire 20.000 banca .... nº .... a favore di Sinclair Computer

Il mio computer è: uno ZX81 ☐ uno Spectrum ☐ altro (specificare) ....

Ho ☐ / non ho ☐ la stampante, ma voglio ☐ comprarla.

Preferisco programmi di gioco ☐, didattici ☐, d'utilità ☐, altro ....

Nome ....

Via .... nº .... CAP. └┴┴┴┴┘ Città ....

Tel. .... Orario ....

Se vuoi collaborare

Registratemi fra i collaboratori regolari di Sinclair Computer.

A titolo di prova vi invio un articolo e la cassetta col programma "....................

...................." di cui vi garantisco l'assoluta originalità autorizzandovene la pubblicazione.

Per il compenso ☐ telefonatemi al .... ☐ scrivetemi all'indirizzo sottoindicato.

Nome ....

Via .... nº .... CAP. └┴┴┴┴┘ Città ....

Tel. .... Orario ....

Se vuoi un consiglio o consigliarci

HELP ....

....

....

....

....

Nome ....

Via .... nº .... CAP. └┴┴┴┴┘ Città ....

Tel. .... Orario ....

Se vuoi votare il software

Questo mese ho comprato i seguenti programmi e li ho giudicati così:

Titolo	Giudizio d'utente (mettere una x nella casella voluta)			
	Ottimo	Buono	Mediocre	Deludente

Nome ....

Via .... nº .... CAP. └┴┴┴┴┘ Città ....

Tel. .... Orario ....

Facoltativo

Da inviare in busta chiusa a:

Sinclair Computer
v.le Famagosta, 75
20142 Milano

Si, voglio abbonarmi

Da inviare in busta chiusa a:

Sinclair Computer
c/o Sigma Informatica
v. Cristoforo Colombo, 49
20090 Trezzano S.N. / Milano

Si, voglio collaborare

Da inviare in busta chiusa a:

Sinclair Computer
c/o Sigma Informatica
v. Cristoforo Colombo, 49
20090 Trezzano S.N. / Milano

Si, chiedo consiglio

Da inviare in busta chiusa a:

Sinclair Computer
c/o Sigma Informatica
v. Cristoforo Colombo, 49
20090 Trezzano S.N. / Milano

Si, voglio votare



ce l'hai?

Il tuo Spectrum è preziosissimo difendilo con la "SUPER GARANZIA"
La Rebit Computer, distributore per l'Italia dei prodotti **SINCLAIR**, ha messo a punto la nuova straordinaria

SUPER GARANZIA

Apri la scatola del tuo **SPECTRUM** acquistato presso un Rivenditore Autorizzato e ci trovi anche un libretto: ti accompagnerà nei tuoi futuri acquisti, dandoti l'occasione per risparmiare oltre 100.000 lire. Ti darà la Garanzia di una perfetta assistenza, e avrai la certezza del valore del tuo autentico **SPECTRUM.** Il libretto della "SUPER GARANZIA" contiene le modalità per l'iscrizione al **SINCLUB**, la federazione di tutti i Sinclair Club Italiani. Inoltre il Coupon sconto per abbonarsi a **"SPERIMENTARE"** il mensile di elettronica che pubblica il bollettino Sinclub: idee, programmi, notizie, vita associativa.

La tessera Software ti da diritto ad uno sconto sull'acquisto dei programmi. Infine nel libretto "SUPER GARANZIA" troverai la possibilità di acquistare la stampante **ZX PRINTER SINCLAIR** ad un prezzo eccezionale.



PER QUESTO UNO SPECTRUM
SENZA LA "SUPER GARANZIA"
E` SOLO UN MEZZO
Spectrum



sinclair
Spectrum
molto di più di una garanzia!!

